Improving the detection of genomic rearrangements in short read sequencing data

Daniel Lee Cameron

orcid.org/0000-0002-0951-7116

Doctor of Philosophy

July 2017

Department of Medical Biology, University of Melbourne
Bioinformatics Division, Walter and Eliza Hall Institute of Medical Research

Submitted in total fulfilment of the requirements of the degree of Doctor of Philosophy

Produced on archival quality paper
Abstract

Genomic rearrangements, also known as structural variants, play a significant role in the development of cancer and in genetic disorders. With the bulk of genetic studies focusing on single nucleotide variants and small insertions and deletions, structural variants are often overlooked. In part this is due to the increased complexity of analysis required, but is also due to the increased difficulty of detection. The identification of genomic rearrangements using massively parallel sequencing remains a major challenge. To address this, I have developed the Genome Rearrangement Identification Software Suite (GRIDSS). In this thesis, I show that high sensitivity and specificity can be achieved by performing reference-guided assembly prior to variant calling and incorporating the results of this assembly into the variant calling process itself. Utilising a novel genome-wide break-end assembly approach, GRIDSS halves the false discovery rate compared to other recent methods on human cell line data. A comparison of assembly approaches reveals that incorporating read alignment information into a positional de Bruijn graph improves assembly quality compared to traditional de Bruijn graph assembly. To characterise the performance of structural variant calling software, I have performed the most comprehensive benchmarking of structural variant calling software to date. This benchmarking highlights the importance not only of read length and coverage size to structural variant callers but also library fragment size, and event size and type. Although no one caller outperforms all others for all data sets, GRIDSS outperforms BreakDancer, CORTEX, CREST, DELLY, HYDRA, LUMPY, manta, Pindel, and Socrates across a wide range of read lengths and fragment sizes once 15x coverage is reached. Finally, I present an exploratory study into the calibration of variant quality scores of structural variant callers. I demonstrate that the quality scores reported by structural variant callers are not well calibrated, and that a model-based calibrated quality score estimator can be constructed from the attributes reported by the variant caller by using long read sequencing of a representative sample. GRIDSS is available at https://github.com/PapenfussLab/gridss and interactive benchmarking results can be found at http://shiny.wehi.edu.au/cameron.d/sv_benchmark.
Declaration

This is to certify that:

(i) the thesis comprises only their original work towards the PhD except where indicated in the preface
(ii) due acknowledgement has been made in the text to all other material used
(iii) the thesis is fewer than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices

Daniel Cameron
Preface

- Chapter 2 includes a preprint of work carried out in collaboration in which 90% is to be considered original work forming part of this thesis. Author contributions are as follows:
  - Daniel L Cameron, Anthony T Papenfuss, Terry P Speed and Jan Schroeder designed the overall approach. Daniel L Cameron designed the assembly approach and implemented GRIDSS. Hongdo Do and Alex Dobrovic provided clinical lung cancer samples and performed DNA preparation; Hongdo Do, Raymar Molania, and Alex Dobrovic performed lung cancer sample data acquisition and predicted lung cancer sample rearrangements; Raymar Molania and Daniel L Cameron performed lung cancer sample variant analysis; Jocelyn S Penington analysed the *Plasmodium falciparum* data. Daniel L Cameron performed all remaining experiments and analysis.

- Chapter 5.1 outlines analysis performed in collaboration with Amit Kumar, John Markham, and Ismael Vergara.

- Funding for this work was provided by the following grants:
  - Australian Postgraduate Award
  - VLSCI PhD Top-Up Scholarship
  - Edith Moffat Scholarship
  - VCCC Picchi Award for Excellence in Cancer Research
Acknowledgements

Foremost, I would like express my gratitude to my supervisor Tony Papenfuss for his unwavering support and assistance, mentorship, and willingness to make himself available for regular discussion despite a busy schedule. Many thanks to my secondary supervisor Terry Speed whose kind and gentle shredding of my occasional statistical abomination has strengthened this thesis immensely.

Thanks are in order for my thesis committee members Gordon Smyth and Melanie Bahlo, and apologies for my last minute scheduling of almost every committee meeting. Thanks to Carolyn de Graaf for expanding my horizons past the purely computation and into the biological world, and Raymar Molania, Hongdo Do, and Alexander Dobrovic for a productive collaboration. I would especially like to thank Jocelyn Penington, Leon di Stefano, Amit Kumar, John Markham, and Ismael Vergara for their testing and feedback of GRIDSS – sorry about the bugs that slipped through to you!

I would like to thank everyone in the Bioinformatics division for their support and collaboration. In particular, I would like to thank Jan Schroeder for getting me up to speed on structural variant calling, Jocelyn Penington, Leon di Stefano, and Alan Ruben for making themselves so freely available for discussions of any nature, and Matthew Wakefield, Alan Ruben, and Melissa Davis for insightful discussions and advice on academic life in general.

Final, in additionally, we would like to thanks my wife Sarah for her her proofreading of the manuscript in it entirely, and correcting myriad mistake of small, so my sentence, am not all like this one. I am indebted to my two little boys and especially Sarah who, despite our agreement to prioritise family over work, still put up with my semi-permanent sleep debt, disappearances overseas, and regular failures to return home at the agreed upon time with relatively good humour. Your encouragement and support over the last four years has been invaluable.
Table of Contents

Abstract ................................................................................................................................................... ii
Declaration ............................................................................................................................................. iii
Preface ................................................................................................................................................... iv
Acknowledgements ................................................................................................................................. v
Table of Contents ................................................................................................................................... vi
Table of Figures ....................................................................................................................................... x
Chapter 1  Introduction ........................................................................................................................ 1
  1.1  Background ............................................................................................................................. 2
  1.2  Genomic Rearrangements ...................................................................................................... 3
  1.3  De novo Assembly ................................................................................................................... 4
  1.4  Variant Detection Approaches ................................................................................................ 5
    1.4.1  Read Alignment ............................................................................................................... 5
    1.4.2  Read Depth (RD) .............................................................................................................. 6
    1.4.3  Discordant Read Pairs (DP) ............................................................................................. 6
    1.4.4  Split Reads (SR) ................................................................................................................ 8
    1.4.5  Assembly (AS) .................................................................................................................. 9
    1.4.6  Combined Methods ...................................................................................................... 10
  1.5  Specialised callers ................................................................................................................. 15
    1.5.1  Repeat-related callers ................................................................................................... 15
    1.5.2  RNA-Seq fusion callers .................................................................................................. 16
    1.5.3  Somatic callers .............................................................................................................. 16
  1.6  Thesis Scope .......................................................................................................................... 17
Chapter 2  GRIDSS: sensitive and specific genomic rearrangement detection using positional de
Bruijn graph assembly ........................................................................................................................... 19
  2.1  Motivation ............................................................................................................................. 19
  2.2  GRIDSS: sensitive and specific genomic rearrangement detection using positional de Bruijn
graph assembly ............................................................................................................................... 21
5.4 Conclusion........................................................................................................................... 118

Chapter 6 Bibliography .................................................................................................................... 120

Appendix A Gene Fusions detected by INTEGRATE and GRIDSS.................................................. 127
Table of Figures

Figure 1-1 A breakpoint is composed of a pair of adjacent break-ends in the rearranged genome .......................................................... 3
Figure 1-2 Types of simple genomic rearrangements .......................................................................................................................... 3
Figure 1-3 Structural variant detection approaches ......................................................................................................................... 5
Figure 2-1 Outline of the GRIDSS pipeline ...................................................................................................................................... 25
Figure 2-2 Variant caller performance on simulated heterozygous genomic rearrangements ..................................................... 28
Figure 2-3 Performance of SV callers on deletion detection in NA12878 at 50x coverage ............................................................. 30
Figure 2-4 Breakdown of GRIDSS variant calling performance ....................................................................................................... 31
Figure 2-5 A tandem duplication identified in a Plasmodium falciparum VAR-gene region ............................................................ 34
Figure 2-6 Comparison of simple event calls by read depth .............................................................................................................. 51
Figure 2-7 Comparison of rearrangement event calls by read depth ............................................................................................... 52
Figure 2-8 F-scores for GRIDSS calls with reciprocal assembly support .......................................................................................... 53
Figure 2-9 F-scores for all GRIDSS calls ........................................................................................................................................ 54
Figure 2-10 GRIDSS precision ......................................................................................................................................................... 55
Figure 2-11 Breakdown of assemblies generated by supporting read count ..................................................................................... 57
Figure 2-12 Breakdown of assembly rates by RepeatMasker annotation .......................................................................................... 61
Figure 2-13 Assembly quality score distribution by repeat type ..................................................................................................... 62
Figure 2-14 Distribution of assembly scores by assembly method and anchor type ........................................................................ 63
Figure 2-15 Assembly rate by anchor type ...................................................................................................................................... 64
Figure 2-16 Variant calls by repeat class ........................................................................................................................................ 65
Figure 2-17 Neochromosome comparison with published call set ................................................................................................. 67
Figure 2-18 NA12878 Precision-Recall curves for multi-mapping aware alignment ........................................................................ 70
Figure 2-19 ROC curve for alternative GRIDSS models ............................................................................................................... 74
Figure 2-20 Breakdown of GRIDSS multi-threaded runtime ......................................................................................................... 77
Figure 2-21 Positional error in variant position relative the simulated event position ........................................................................ 78
Figure 2-22 Breakdown of memory usage during assembly ............................................................................................................... 80
Figure 3-1 Effect of depth of coverage on variant caller performance ............................................................................................. 89
Figure 3-2 Effect of read length on variant caller performance ...................................................................................................... 90
Figure 3-3 Effect of fragment length on variant caller performance ............................................................................................... 91
Figure 3-4 Variant caller performance on human cell line deletions ............................................................................................... 97
Figure 3-5 Average variant caller runtime performance on simulated data sets ........................................................................... 98
Figure 4-1 BreakDancer Precision-recall and calibration curves .................................................................................................... 105
Figure 4-2 CREST Precision-recall and calibration curves ............................................................................................................. 106
Figure 4-3 DELLY Precision-recall and calibration curves .............................................................................................................. 107
Figure 4-4 GRIDSS Precision-recall and calibration curves ................................................................. 108
Chapter 1  Introduction

Although the detection of genomic rearrangements in short read sequencing data is conceptually simple, subtle complexity and noisy data make accurate and specific detection of these events in real data a challenging open problem. In this thesis, the current state-of-the-art approaches are outlined and evaluated, and a novel approach to the detection of genomic rearrangements is presented.

In this introductory first chapter, the basics of short read sequencing and genome rearrangements are introduced. After a brief background on de novo assembly, the high-level approaches to genomic rearrangement detection are detailed and existing approaches and implementations are reviewed.

Chapter 2 presents the two key novel contributions of this thesis to genomic rearrangement detection. Firstly, I show that performing reference-guided genome-wide break-end assembly of all available supporting reads prior to variant calling improves variant calling compared to existing approaches. Secondly, I show that encoding to reference alignment constraints within the assembly graph itself makes the assembly more robust to repetitive sequence than windowed, targeted or de novo de Bruijn graph assembly. This chapter includes details and benchmarks of the implementing software GRIDSS: the Genomic Rearrangement IDentification Software Suite.

Chapter 3 presents the results of the most comprehensive benchmarking of general-purpose structural variant calling software to date. As well as demonstrating that GRIDSS outperforms existing state-of-the-art approaches once 15x coverage is reached, this benchmark identifies conditions in which several widely used callers exhibit unexpectedly poor performance.

Chapter 4, the final results chapter, attempts to address a problem in the downstream interpretation of structural variant calls. Knowing the estimated probability of a given variant call being correct is important when performing variant analysis. Unfortunately, whilst well-calibrated probabilities can be obtained for single nucleotide and other small variants, this information is lacking from structural variant calls. This chapter contains an exploratory study into the generation of well-calibrated probability-based quality scores from the output of any structural variant caller.

Finally, in the concluding chapter, the existing usages of GRIDSS are outlined as a demonstration of the utility of GRIDSS in both research and clinical settings. The future of genomic rearrangement detection is discussed in the light of new technology and algorithms, and potential further work adapting GRIDSS to incorporate these new advances is discussed. The thesis concludes with a final summary of the contributions made by the research embodied within this dissertation.
1.1 Background

The advent of high-throughput sequencing technologies has enabled genomic research on a scale not previously possible. Whilst the human genome project completed the sequencing of the first mammalian genome in 2003 at a cost of over USD$2.7 billion [1], by 2017, the cost of human genome resequencing was as low as USD$1,000. Currently, the cheapest and most prevalent sequencing technologies are short read sequencing technologies such as the Illumina NGS platform in which millions of DNA fragments are sequenced in parallel. Although read lengths from short read sequencers have improved over an order of magnitude in the last decade (from 25bp to 2x300bp in the case of the Illumina sequencing by synthesis platform), the short read lengths continue to pose challenges for sensitive and specific detection of genomic rearrangements. As the cost of sequencing has reduced at a rate faster than Moore’s Law, the cost of the associated computation and analysis has become increasingly relevant and the need for tools with improved performance, both in terms of computational cost and quality of results, in analysis pipelines is greater than ever.

In a typical resequencing pipeline, sequenced reads are aligned to a reference genome with variant calling performed on the aligned reads. Depending on the workflow requirements, several additional steps can be performed including marking/removing duplicates, trimming adapter sequence, and base quality score recalibration. For an Illumina sequencer using paired-end sequencing, two FASTQ [2] files are output, the first containing the first read of each pair, and the second containing the second read of each pair.

Early aligners designed for short reads in the 30-40bp range such as MAQ [3] performed un-gapped alignment, tolerating only 1-2 base mismatches in the alignment. As sequencing technology improved and read lengths increased, aligners allowing more relaxed alignment criteria were developed, with aligners such as bwa [4] and bowtie2 [5] not only allowing gapped (Needleman–Wunsch [6]) alignments, but also allowing local (Smith-Waterman [7]) alignments in which clipped’ bases at the ends of the read can be “soft-clipped” and excluded from alignment. Determining the correct mapping location of reads originating from repetitive sequences remains challenging and while aligners typically reporting only the single best mapping location, most can be configured to report multiple locations with aligners such as mrFast [8] specifically designed for this purpose.

Whilst single nucleotide variants (SNVs) can be determined directly from the alignment of the read against the reference genome, only small insertions and deletions (indels) can be detected directly from read alignment as above a threshold size (typically less than half the length of the read), the correct alignment will be sufficiently low scoring that it will not be reported. As a result, the detection methods used to detect SNVs and small indels can differ greatly from those used to detect larger
structural events. This distinction is reflected not only in the analysis pipelines, but also the variant databases where 50bp is recommended as the cut-off length for submission to either dbSNP [9] or dbVar/DGVa [10].

1.2 Genomic Rearrangements

The term structural variation has been used to describe events ranging from single base pair insertions or deletions, all the way up to whole chromosome events [11]. Although standardisation efforts like the MISO Sequence Ontology database [12] provide definitions of terms such as “structural variant”, ambiguity in usage of such terminology still remains. This thesis uses the terms structural variant and genomic rearrangements synonymously and terminology used in the VCF file format specifications [13] has been adopted. Specifically, “an arbitrary rearrangement event can be summarized as a set of novel adjacencies. Each adjacency ties together 2 break-ends” and a breakpoint is defined as the novel adjacency between two break-ends (Figure 1-1).

![breakpoint diagram](image)

**Figure 1-1** A breakpoint is composed of a pair of adjacent break-ends in the rearranged genome

Between the two break-ends, a breakpoint can contain any number of additional bases. These bases are referred to as non-template sequence insertions. Using this approach, both simple rearrangements such as insertions, deletions, inversions, tandem duplications, and translocations (Figure 1-2), as well as complex rearrangements such as those resulting from chromothripsis can be broken down into their constituent breakpoints.

![rearrangement types diagram](image)

**Figure 1-2** Types of simple genomic rearrangements
Notably this definition of structural variation requires the presence of at least one breakpoint thus excludes whole chromosome events such as aneuploidy in which a change in chromosomal copy number occurs without the introduction of any additional breakpoints.

1.3 De novo Assembly

Before describing the signals of genomic rearrangements in short read sequencing data and the approaches used for finding them, it is helpful to introduce de novo assembly. In the absence of a reference genome to which reads can be aligned to, reconstruction of a genome from sequencing data requires the assembly of the sequencing reads de novo. As high-throughput “shotgun” sequencing technologies sequence many random fragments of the source DNA, reconstructing the originating sequence is a non-trivial problem. To solve this problem, two distinct computational approaches to genome assembly have been developed [14].

The first approach to genome assembly uses a three step overlay-layout-consensus (OLC) process. In the first step, the pair-wise overlap between all reads is calculated. From these overlaps, a layout graph is generated with nodes representing reads with an edge present between each pair of overlapping reads. Using the sequence and overlap information in the graph, consensus sequence contigs are generated. This style of approach, first used during the sequencing of the ΦX174 bacteriophage [15], is suitable for sequencing technologies such as Sanger sequencing in which the reads are relatively long (200bp+), and whose sequencing error profile contains relatively high indel error rate.

The second approach to genome assembly is based on splitting reads up into kmers—sub-sequences of \( k \) bases. The kmers (and their reverse complements) from all reads are, after error correction based on kmer frequency, inserted into a de Bruijn graph. Each sequence kmer is represented by a node in the De Bruijn graph. An edge is present between nodes \( a \) and \( b \) if the final \( k - 1 \) bases of \( a \) match the first \( k - 1 \) bases of \( b \). After error correction of bubbles and leaves caused by sequencing errors, a contig is called for each unbranched path through the de Bruijn graph. De Bruijn graph assembly is more scalable than OLC assembly as it omits the quadratic time pair-wise alignment required for OLC assembly. As a result, de Bruijn graph assembly has been the primary assembly approach used for whole genome shotgun assembly of large genomes. Whilst this kmer based approach is suitable for short read sequencing technologies such as Illumina, in which sequencing errors are primarily substitution errors, it is problematic for long read sequencing technologies such as PacBio and Oxford Nanopore, due to their high indel error rates.
For pair-end sequencing data, the position and orientation of the reads provides additional information that an assembler can use when reconstructing the genome. The process of joining assembly contigs based on read pair information is called scaffolding, with numerous scaffolding tools available [16]. These tools enable improved assembly as they allow assembling across sub-sequences in which coverage is insufficient for contig assembly, as well across short (with respect to the fragment size distribution of the library used for scaffolding) repetitive sequences.

1.4 Variant Detection Approaches

A number of approaches have been taken to detect structural variants from sequencing data. Broadly speaking, these are based on identifying variants from read alignments, read depth (RD), discordantly mapped read pairs (DP), split read analysis (SR) using one end anchored (OEA) read pairs or soft-clipped (SC) reads, or assembly (AS) contigs. Each of these approaches has distinct advantages and limitations.

![Diagram showing variant detection approaches](image)

**Figure 1-3 Structural variant detection approaches**

1.4.1 Read Alignment

Read alignment based approaches identify variants directly from the alignment of reads. Aligners outputting in the standard SAM/BAM [17] file format report alignments using an alignment starting location and a CIGAR (Concise Idiosyncratic Gapped Alignment Report) string indicating the alignment of each read base relative to the reference sequence. In conjunction with the reference sequence, this alignment information allows for the identification of SNVs and small insertions and deletions (indels). Due to the nature of alignment scoring, the maximum size of the indel that can be detected using read alignment directly is limited to approximately half the read length. Although performing local
assembly of candidate indels can extend the maximum detectable event size [18, 19], such variant callers are typically specialised small indel or SNVs/small indel callers and are unable to call other types of structural variation. Due to the nature of the event signal, detection methods used to detect SNVs and small indels differ from those used to detect large structural events. This distinction is reflected not only in the analysis pipelines, but also the variant databases where 50bp is recommended as the cut-off length for submission to either dbSNP [9] or dbVar/DGVa [10].

1.4.2 Read Depth (RD)

Approaches based on read depth detect copy number variation (CNV) based on read mapping rate. Due to the similarities between sequencing read depth and microarray intensity signals, the techniques developed for RD CNV analysis are similar to those developed for microarray CNVs. Unlike the other approaches, read depth approaches detect deletion or duplication of DNA segments directly. A typical CNV detection algorithm will segment the genome into regions of identical copy number based on the GC-adjusted average read depth (typically using a 1 kilobase bin size) and, for some callers, single nucleotide variant allele fraction of SNVs within each region.

Since it is the DNA segment itself that is detected, large events can be reliably detected but smaller events such as those under 1kb are generally not detected by CNV callers due to technical variation and stochastic noise dominating the signal at low read counts. Unfortunately, copy number neutral events such as inversions do not change the copy number and are unable to be detected by RD based methods. Furthermore, since breakpoints are not identified, the locations of duplicated segments are unknown and RD based approaches are unable to distinguish between tandem duplication and translocations. As expected, novel sequence insertions of any size are unable to be detected.

In this thesis, approaches such as read depth based CNV detection algorithms, which are unable to place variants or determine breakpoint partner locations, are considered a separate problem class and are examined only when the technique is combined with an algorithm which is able to determine breakpoint partner locations. Comparisons of CNV detection software can be found in existing review literature [20-22].

1.4.3 Discordant Read Pairs (DP)

Discordant read pair based methods identify structural variants by comparing the alignment strands and positions of each read pair to the overall distribution of the input library. Read pairs mapping unexpectedly close together, far away, or in an unexpected orientation are considered discordant read pairs. Such read pairs could be caused by the reads originating from an unexpectedly large or small DNA fragment, one or both of the reads being incorrectly placed by the aligner, chimeric fragments
caused by the ligation of two independent DNA fragment during library preparation, or an underlying structural variant spanned by the fragment. Structural variants are called based on the identification of clusters of these discordant read pairs.

The most common method for identification of discordant read pairs is to use a fixed window in which all reads within a size range are considered concordant. This window can either be supplied explicitly by the user (e.g. VariationHunter [23], HYDRA [24]), or based on a multiple of the standard deviation (e.g. BreakDancer [25]) or median absolute deviation [26] of the library fragment size distribution. The method of discordant read pair clustering varies considerably between variant callers. The simplest approach uses simple greedy clustering, either adding to existing read clusters (e.g. HYDRA [24], Meerkat [27], SVMiner [28]), or at an even courser resolution of fixed-width genomic bins (e.g. SVDetect [29]). A more theoretically sound approach enumerates all maximal cliques (e.g. GASV [30], VariationHunter [31]), that is, all maximal subsets of mutually consistent discordant read pairs. These cliques are either reported directly followed by downstream filtering of similar events (e.g. GASV [30]), or used to disambiguate multi-mapping reads (VariationHunter [23], GASVPro [32]). As using a fixed threshold creates a hard limit on the smallest event size detectable, some callers improve small event detection capability by incorporating the discordant read pair identification into the clustering itself (e.g. CLEVER [33]), or fit the local distribution of fragment sizes to a pair of reference/variant distribution models directly (e.g. MoDIL [34]).

DP analysis has a number of distinct drawbacks and advantages. Most obviously, this technique is only applicable to pair-end sequencing technology. Secondly, as the inferred breakpoint occurs in the unsequenced portion of the fragment, the exact breakpoint location cannot be identified. Thirdly, small indels are unable to be identified as the minimum event size is limited by the width of the fragment size distribution. Deeper coverage will reduce the width of the breakpoint confidence interval but the exact breakpoint sequence will remain unknown in the general case. These disadvantages are mitigated by advantages of this approach. Since the read sequence itself is only required for determining the alignment location of the read, DP approaches are applicable to reads of any read length. This was particularly important for first generation high-throughput sequencers where the 36bp read lengths severely limited the detection capability of split read and read alignment based approaches.

Unlike read-based approaches whose detection capability is independent of library fragment size, read pair based approaches are highly dependent on fragment size. Increasing the fragment size raises the average number of read pairs spanning each structural variant and thereby increases the signal strength at a given coverage depth. However, this occurs at the cost of increased breakpoint
imprecision for each read pair, and reduced small event detection capability, as the deviation of the library fragment distribution also increases with fragment size.

In degraded DNA scenarios, such as that found in Formalin-Fixed, Paraffin-Embedded (FFPE) samples, the signal strength of DP based approaches is weakened due to the short mean fragment size. For non-degraded DNA, the DP signal strength for structural variation can be dramatically improved through the use of mate-pair sequencing [35]. This alternate library preparation protocol incorporates an additional circularisation and fragmentation step which, rather than sequencing both ends of 200-500bp fragments in an inward-facing ‘forward-reverse’ FR orientation, results in the sequencing of the ends of 2,000-20,000bp fragments in ‘reverse-forward’ RF orientation. Due to the library preparation protocol used, sequencing of mate-pair libraries results in sequencing data containing a mix of pair-end and mate-pair reads. This complication requires either the variant caller to explicitly handle libraries with mixed insert size distributions and read orientations, or additional pre-processing of the library to deconvolute the reads (e.g. NxTrim [36]).

1.4.4 Split Reads (SR)

Split read based approaches identity structural variants by partial alignment of a single read to two separate locations. If a read spans a breakpoint, part of the read will align to one break-end, with the remainder of the read at the other break-end. Split reads can be identified in a number of ways. For aligners such as bwa [4] that directly report split alignments, identification of split reads from the aligned reads is a trivial operation. For aligners that do not directly report split alignments, identification of split reads depends on whether the aligner performs local alignment, where the aligner will partially align a read, or global end-to-end alignment, where the aligner requires all read bases to be aligned to the reference. In the case of local alignment, the unaligned ‘soft-clipped’ (SC) bases can be realigned back to the reference to identify split reads. For global end-to-end alignment, split reads will fail to fully align to either mapping location and will remain unmapped. For paired-end sequencing, this results in a one-end anchored (OEA) read pair in which only one read in the pair is mapped. In such scenarios, split reads are identified by first identifying a partial alignment in a location and orientation consistent with the expected fragment size distribution of the library, followed by identification of the mapping location of the remainder of the read (e.g. Pindel [37]).

Since the structural variant breakpoint occurs within the read, split read analysis allows for the determination of the exact breakpoint sequence. Unfortunately, since the read is split into two shorter subreads, unambiguous placement of these subreads is difficult. For short read data, subread alignment against repetitive genomes such as the human genome is so difficult that many split-read approaches restrict the candidate alignment locations to reduce the false positive and multi-mapping
rates. The candidate alignment locations are either restricted to events smaller than a threshold size (e.g. SVSeq [38], Pindel [37]), to locations with discordant read pair support (e.g. SVSeq2 [39], SoftSV [40]), or to either (e.g. PRISM [41], VarDict [42]). As the read lengths generated from high-throughput sequencers lengthened, unconstrained split read realignment became a viable option (e.g. bresoq [43], ClipCrop [44], CREST [45], Socrates [46], SRiC [47]).

1.4.5 Assembly (AS)

Assembly has been used in the identification of structural variants using a range of approaches. Broadly speaking, these approaches can be categorised as de novo assembly, breakpoint assembly, and break-end assembly.

The de novo assembly approach assembles reads into longer contiguous sequences (contigs), then uses read alignment and/or split read analysis of the contigs to identify variants. No read alignment is performed prior to assembly thus the only assembly constraints are the read bases themselves. Contigs are aligned back to the reference genome using a standard short read aligner (e.g. laSV [48]), or a sequence aligner (e.g. BLASTZ [49] or LASTZ [50]) designed for long genomic sequences (e.g. CORTEX [51], SOAPsv [52], AsmVar [53]). After alignment, the same techniques used by read alignment and split read callers are used for the actual variant calling.

As any read could potentially be assembled with any other read, de Bruijn graph based assembly approaches (e.g. cortex, Velvet, ABySS [51, 54, 55]) are used for contig generation as the quadratic time complexity of overlap-layout-consensus (OLC) assembly is computationally infeasible for mammalian genomes. An important differentiator between de novo assembly and other approaches is the ability to detect large novel insertions. Since other methods rely on read alignment, their novel sequence detection capability is limited by the read length (read alignment/SR) or fragment size (DP) or not at all (CNV).

An alternative assembly approach that does not have the high computational cost associated with de novo assembly, is to use breakpoint assembly as a validation step. Candidate variants are first identified from read alignment (e.g. GATK [56]), SR, DP (e.g. SoftSV [40]), or the output of other variant callers (e.g. SVMerge [57], MetaSV [58]). For each candidate breakpoint, targeted assembly across both break-ends is performed using either all reads [56-59], or just the variant-supporting subset [40]. Since many independent assemblies are performed each using only a handful of reads, both de Bruijn graph and OLC assembly approaches are viable and have been used to generate assembly contigs [40, 59]. The purpose of performing this assembly is two-fold: the lack of assembly support can be used to filter low quality false positive variants and the assembly contigs can be used to refine inexact DP calls.
and determine the exact breakpoint sequence including any non-templated bases inserted at the breakpoint.

If detection is restricted to small events, genome wide breakpoint assembly can be performed without prior identification of candidate variants or loci. This windowed assembly approach divides the genome into fixed-size windows (ranging from tens (e.g. SOAPindel [19]) to hundreds (e.g. DISCOVAR [60]) of kilobases) and performs independent assembly on all reads and read pairs mapping to those windows before performing targeted variant calling within each window. Whilst this approach is faster than full de novo assembly, only events smaller than the window size are detectable and like the other alignment-based approaches, it cannot fully reconstruct novel insertions larger than twice the library fragment size as unmapped read pairs are not considered in windowed assembly.

The final use of assembly by structural variant callers is to perform targeted break-end assembly prior to variant calling. This approach has been used in two separate contexts. Firstly, targeted break-end assembly of SC read clusters has been used to perform split read identification on the resultant contigs (e.g. CREST [45, 61]). Secondly, targeted break-end assembly of OEA clusters has been combined with de novo assembly of unmapped reads to detect large novel insertions (e.g. NovelSeq [62]).

1.4.6 Combined Methods

With each of these approaches having distinct drawbacks and advantages, tools combining multiple approaches are becoming increasingly common with a number of combination strategies emerging. The most prevalent approach is to use identify split reads in regions supported by DP analysis [27, 38-41, 63]. Since target regions are identified through DP support, variants with strong SR support but insufficient DP support will not be identified. To avoid this, split reads can be independently identified and clustering performed on SR and DP evidence combined [64, 65]. Read depth information has been combined with most combinations (RD/DP [32, 66, 67], RD/SC [43, 68], RD/SC/DP [69]), but, at present, read depth information has not been directly combined with assembly as the primary variant calling signals.

An alternative approach to combination is through the use of a meta-caller. These meta-callers combine the results of variant callers either directly, such as done in the ICGC-TCGA DREAM Somatic Mutation Calling Meta-pipeline Challenge [70], or in conjunction with targeted breakpoint assembly to validate putative calls (e.g. SVMerge [57], MetaSV [58], Parliament [71]). By decoupling the constituent variant callers from the meta-callers, advances in any one of the callers used, as well as additional callers, can be incorporated in a relatively straight-forward manner.
<table>
<thead>
<tr>
<th>Software</th>
<th>Input format</th>
<th>Output format</th>
<th>DP</th>
<th>OEA</th>
<th>SC</th>
<th>Read Depth</th>
<th>Assembly</th>
<th>Multi-mapping aware</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VariationHunter[31]</td>
<td>DIVET</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td>Chooses mapping locations of multi-mapping reads such that the number of SV calls is minimised.</td>
</tr>
<tr>
<td>GASV[30]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Computes all mutually consistent DP subsets (maximal cliques) based on fragment size threshold.</td>
</tr>
<tr>
<td>Pindel[37]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td>Pattern growth algorithm finds split read mapping nearby anchoring mate read. Event size limited by search window size.</td>
</tr>
<tr>
<td>Breakdancer[25]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BreakDancerMax identifies regions containing more DPs than empirically expected. BreakDancerMini calls small indels from read pairs BreakDancerMax considered concordant.</td>
</tr>
<tr>
<td>SVDetect[29]</td>
<td>BAM</td>
<td>BED</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Uses sliding window to perform DP clustering.</td>
</tr>
<tr>
<td>SVMerge[57]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Targeted assembly validation of calls from BreakDancer, Pindel, SE Cluster, RDXplorer, and RetroSeq.</td>
</tr>
<tr>
<td>SOAPsv[52]</td>
<td>FASTQ</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reference alignment of whole genome de novo assembly contigs. Homozygous variants &lt; 50kbp only.</td>
</tr>
<tr>
<td>SRiC[47]</td>
<td>FASTQ</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split read identification through BLAT gapped alignments. Support threshold varies with indel size to control FDR.</td>
</tr>
<tr>
<td>Software</td>
<td>Input format</td>
<td>Output format</td>
<td>DP</td>
<td>OEA</td>
<td>SC</td>
<td>Read Depth</td>
<td>Assembly</td>
<td>Multi-mapping aware</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------------</td>
<td>----</td>
<td>-----</td>
<td>----</td>
<td>------------</td>
<td>----------</td>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CREST[45]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Targeted CAP3 break-end assembly of SC clusters; BLAT alignment of assembled contigs.</td>
</tr>
<tr>
<td>SVseq[38]</td>
<td>SAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split read caller. FDR reduced by requiring DP support. Event size limited by search window size.</td>
</tr>
<tr>
<td>CommonLAW[72]</td>
<td>DIVET</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Multi-sample extension of VariationHunter.</td>
</tr>
<tr>
<td>ClipCrop[44]</td>
<td>BAM</td>
<td>BED</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split reads identified by bwa alignment of SC bases</td>
</tr>
<tr>
<td>GASVPro[32]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Combines GASV calls with read depth signal. MCMC placement of multi-mapping reads.</td>
</tr>
<tr>
<td>SVseq2[39]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split read mapping of soft clipped reads to only regions with DP support.</td>
</tr>
<tr>
<td>SVM2[73]</td>
<td>Unknwn</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Support vector machine used to call DP events. SVM trained by on simulated indels. Software no longer available.</td>
</tr>
<tr>
<td>PRISM[41]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split read mapping to regions with DP support, or small indels.</td>
</tr>
<tr>
<td>DELLY[63]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Breakpoint position of DP calls refined by searching for supporting split reads.</td>
</tr>
<tr>
<td>CLEVER[33]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Clustering of all read pairs to identify discordant clusters.</td>
</tr>
<tr>
<td>SVM2miner[28]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DP clustering with event type uncertainty.</td>
</tr>
<tr>
<td>cortex_var[51]</td>
<td>FASTQ</td>
<td>VCF</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>De novo assembly, LASTZ mapping of contigs</td>
</tr>
<tr>
<td>BreakPointer[74]</td>
<td>Custom</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Breakpoint position of DP calls refined by searching for supporting split reads. Requires BAM &amp; dRanger rearrangement predictions as input.</td>
</tr>
</tbody>
</table>

---

Page 12
<table>
<thead>
<tr>
<th>Software</th>
<th>Input format</th>
<th>Output format</th>
<th>DP</th>
<th>OEA</th>
<th>SC</th>
<th>Read Depth</th>
<th>Assembly</th>
<th>Multi-mapping aware</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SV-M[75]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SVM trained sanger validation. Heterozygous events only. Event size limited to &lt; 5kbp.</td>
</tr>
<tr>
<td>PeSV-Fisher[66]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Greedy DP clustering. RD used for variant classification and annotation.</td>
</tr>
<tr>
<td>Bellerophon[76]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Soft clipped reads refine DP clusters. Interchromosomal only.</td>
</tr>
<tr>
<td>Meerkat[27]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Greedy DP clustering. BWA/BLAST split read identification.</td>
</tr>
<tr>
<td>SoftSearch[65]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Combined cluster of SR and DP.</td>
</tr>
<tr>
<td>Socrates[46]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Split reads identified by bowtie2 alignment of SC cluster consensus sequences.</td>
</tr>
<tr>
<td>LUMPY[69]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Signals converted to breakpoint probability distributions then combined.</td>
</tr>
<tr>
<td>SVFinder[77]</td>
<td>BAM</td>
<td>BED</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Greedy DP clustering.</td>
</tr>
<tr>
<td>Gustaf[78]</td>
<td>FASTQ</td>
<td>VCF</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Multi-mapping split read alignment.</td>
</tr>
<tr>
<td>TIGRA-ext[59]</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Targeted assembly validation of calls from BreakDancer, Pindel, or DELLY.</td>
</tr>
<tr>
<td>llaSV[48]</td>
<td>FASTQ</td>
<td>VCF</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>De novo assembly, bwa mapping of contigs. Validational bwa alignment of reads to reference + putative SVs.</td>
</tr>
<tr>
<td>Software</td>
<td>Input format</td>
<td>Output format</td>
<td>DP</td>
<td>OEA</td>
<td>SC</td>
<td>Read Depth</td>
<td>Assembly</td>
<td>Multi-mapping aware</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------------</td>
<td>----</td>
<td>-----</td>
<td>----</td>
<td>------------</td>
<td>----------</td>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RAPTR-SV[79]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Uses VariationHunter clustering and multi-mapping resolution. SR identification using MrsFAST alignment of 50/50 split of read bases.</td>
</tr>
<tr>
<td>MetaSV[58]</td>
<td>BAM</td>
<td>VCF</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td>Targeted assembly of calls from BreakDancer, Pindel, CNVnator, BreakSeq, and soft clips.</td>
</tr>
<tr>
<td>BreaKmer[80]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td>Targeted assembly of misaligned reads. Not suitable for whole-genome.</td>
</tr>
<tr>
<td>SoftSV[40]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td>Targeted OLC assembly of SC reads. Target regions identified by DP clusters.</td>
</tr>
<tr>
<td>Hydra-Multi[81]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td>Multi-sample extension of HYDRA.</td>
</tr>
<tr>
<td>SV-Bay[67]</td>
<td>BAM</td>
<td>Custom</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Bayesian model using read depth, mappability, and GC bias to filter candidate DP clusters.</td>
</tr>
<tr>
<td>GRIDSS</td>
<td>BAM</td>
<td>VCF</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Whole genome break-end assembly. Clustering of contigs, SR, DP.</td>
</tr>
</tbody>
</table>

Table 1 Summary of publicly available general purpose structural variant calling software
1.5 Specialised callers

For some types of structural variation, specialised callers are required for reliable identification of event of that type. Such edge cases include repetitive sequence, novel insertions such as viral integration, gene fusion detection, and somatic variant identification. Two classes of specialised callers that have seen significant development of specialised detection strategies are gene fusion detection from RNA-Seq data, and repetitive element detection.

1.5.1 Repeat-related callers

Repetitive sequences are DNA sequences that occur in multiple locations in the reference genome. These repeats can range from single base pair homopolymer runs, to larger centromeric repeats and interspersed repeats hundreds to a few thousands of base pairs in length, to large segmental duplications up to hundreds of kilobases in length. Whilst segmental duplications can be handled by generic structural variant callers, interspersed elements and short tandem repeats are problematic as alignment of reads originating from repetitive elements results in many candidate mapping locations.

Most generic structural variant callers are unable to handle reads mapping equally well to multiple mapping locations. Such multi-mapping reads are typically filtered before variant calling to reduce the false positive rate. This loss of information results in an increased false negative rate in repetitive regions of the genome, a problematic loss given that over 50% of the entire human genome can be associated with a repetitive element [82]. Whilst not all repetitive elements correspond to unmappable regions, these are highly correlated. Repetitive element mappability improves with read length, and by 75bp over 80% of the human genome is uniquely mappable [83]. A number of callers address multi-mapping reads either through scoring all potential variants (e.g. GASV [30]), greedy clustering of potential mapping locations (e.g. Meerkat [27]), or by assigning mapping location such that the total number of variant calls is minimised (e.g. VariationHunter, RAPTR-SV [23, 79]).

Short tandem repeats (STRs), also known as microsatellites, consist of a tandem repetition of a short DNA sequence (typically 2-6 nucleotides). Due to the combination of poor mappability and increased polymerase stutter in STRs, specialised tools have been developed for STR detection (e.g. lobSTR, repeatseq, STRViper [84-87]).

A third class of repeat-related callers are the transposable element callers. These callers use alignment to a database of known repeat elements [88, 89] to identify insertions, deletions, and translocation of transposable elements (e.g. RetroSeq, tea [90, 91]). A similar database-based approach is used for detection of viral insertions (e.g. VirusSeq, VirusFinder, ViralFusionSeq [92-94]). If the variants of
interest are already known, targeted variant calling can be performed only on the regions of interest (e.g. BreakSeq, BreakSeq2 [95, 96]) thus reducing the computational burden. Whilst this approach is useful for short reads not able to be mapped using split read approaches, it requires a catalogue of candidate event and is only suitable for detection of known events.

1.5.2 RNA-Seq fusion callers

Unlike whole genome sequencing (WGS) in which the DNA is sequenced directly, whole transcript sequencing (WTS), also known as RNA-Seq, uses high-throughput sequencing to sequence the relative abundances of expressed RNA transcripts, typically through reverse transcribing to cDNA [97]. As it is the mRNA transcript that is being sequenced, WTS is different to WGS in two key aspects: Firstly, read depth varies greatly as it is determined by relative transcript abundance and not directly by copy number. Secondly, exon-exon junctions result in orders of magnitude more apparent structural variants when aligned to the reference genome. To handle these differences, and improve gene fusion detection, a number of specialised WTS structural variant callers have been developed. Overall, the same approaches used for genomic structural variant detection are used for gene fusion detection, with annotations of known transcripts providing additional information. Alignment is typically performed by a junction-aware aligner able to directly align split reads across exon-exon junctions (e.g. STAR, TopHat2, HiSAT [98-100]), with variants called from DP (e.g. ChimeraScan [101]), SR/DP support (e.g. TopHat-Fusion, FusionFinder [102, 103]), or from de novo transcript assembly (e.g. EBARDenovo, trinityrnaseq [104, 105]). If both WTS and WGS data is available, specialised callers identify gene fusions and their corresponding genomic breakpoint, using DP (e.g. nFuse, Comrad [106, 107]), DP/SR (e.g. INTEGRATE [108]) or meta-calling approach (e.g. [109]) combining generic SV callers with WTS gene fusion callers.

1.5.3 Somatic callers

A number of approaches to somatic variant detection have been used. Most basically, variant calling can be independently performed on the tumour and normal and variants filtered to remove germline calls. More sophisticated analysis combines the tumour and normal thus allowing for the correct classification of variants with insufficient support to identify in a single sample [69]. An alternative approach available only for somatic calling is to detect somatic calls directly from the read sequences without prior alignment (e.g. SMUFIN [110], novoBreak [111]). Like de novo assembly, this approach does not rely on alignment to the reference genome and as a result shares similar advantages and disadvantages: namely the capability to detect a wider range of events at the cost of limited detection capability in repetitive regions.
1.6 Thesis Scope

When viewing the landscape of genomic rearrangement using short read sequencing data, several gaps emerge.

Firstly, although many tools and algorithms have been developed using coverage, split read, read pair, and assembly-based approaches, there remain some under-developed areas. Of the three assembly approaches, only de novo and breakpoint assembly are well covered. To date, the break-end assemblers developed have utilised either read pair or split read information, but not both. Furthermore, the existing breakpoint and break-end assemblers use the reference alignment merely to limit the reads considered for assembly, thus discarding read alignment constraints that could be incorporated into the assembly itself.

Secondly, there is a lack of empirically justified guidance regarding which structural variant calling approaches and tools are appropriate for which data sets. Whilst a number of review papers comparing structural variant calling approaches have been published [82, 112-119], they are based on theoretical expectations. Such publications provide recommendation based purely on whether an algorithmic approach is capable of detecting events of a certain type and size. Practical considerations, such as choosing between different implementation of the same fundamental approach, and which tools focus on sensitivity vs specificity, require empirical data. Benchmarking studies have been limited to SNV/small indel callers [120-125], or CNVs [20], with the exception of comparisons included in the publication of new methods. Such comparisons invariably report results favourable to the tool being published.

Finally, there is a lack of consistency in quality scores assigned to variants. Whilst SNV callers universally report phred-scaled quality scores representing the probability of the variant call being correct, SV callers do not. Most SV callers merely report counts of support reads and the quality scores of those that do are not well-calibrated.

This thesis attempts to address these issues. In Chapter 2, GRIDSS: the Genomic Rearrangement IDentification Software Suite is introduced. As well as several utility programs useful for handling structural variants, this software contains a novel break-end assembler and structural variant caller. This assembler fills the gap not covered by existing methods by performing genome-wide break-end assembly of not only both split reads and discordant read pairs, but of all reads providing SV support. Using a positional de Bruijn graph approach not previously shown to be computationally feasible at genome scale, this assembler demonstrates that by incorporating read alignment information into the assembly itself, the assembler can be made more robust to repetitive sequence than traditional de
Bruijn graph assembly. The GRIDSS structural variant caller combines split read and read pair evidence with the assembly contigs in a probabilistic model. Human cell line results are shown demonstrating this novel approach results in a false discovery rate half that of existing approaches without loss of sensitivity.

Chapter 3 details the results from performing the most comprehensive benchmarking of structural variant callers to date. In this chapter, a cross-section of well-cited structural variant callers representative of the diversity of detection methods are evaluated on both simulated and human cell line data. By simulating across a wide range of read lengths, depths of coverage, fragment sizes, event type and event sizes, the detection limits and capabilities of the callers is outlined. Using human cell line data, I show that performance data simulated from a reference genome represents an optimistic upper bound that is not recapitulated on real data. The chapter concludes with practical recommendations regarding the criteria for selection of appropriate structural variant detection software for a given data set.

Chapter 4 describes an initial exploration of well-calibrated structural variant quality scores. This chapter demonstrates that linear models using variant caller-reported annotation can be used to calculate a calibrated quality score by long read sequencing of a representative sample.

In the concluding chapter, the utility of GRIDSS in both research and clinical setting is demonstrated through the outlining of a number of projects employing GRIDSS. Finally, the future of genome rearrangement detection is discussed in the light of algorithmic and technological advances, and proposals regarding how these techniques and technologies can be combined with those introduced in this thesis are outlined.
Chapter 2  GRIDSS: sensitive and specific genomic rearrangement detection using positional de Bruijn graph assembly

This chapter introduces GRIDSS (the Genomic Rearrangement IDentification Software Suite—a software toolkit and new method for detecting genomic rearrangements. The chapter is an extended version of the GRIDSS preprint [126]. Some context and background information is provided first (Section 2.1). This is followed by the preprint in Section 0, and Supplementary Materials in Section 2.3. Contained in this chapter but not present in the preprint are implementation details and results of a multi-mapping aware version of GRIDSS designed for improved sensitivity in repetitive genomic regions.

2.1 Motivation

Broadly speaking, general purpose structural variant calling software uses split read analysis, discordant read pair analysis, assembly, read depth information, or a combination of these approaches to identify genomic rearrangements. To date, over 50 such tools have been published (Table 1), and whilst split reads, discordant read pairs, or a combination have been extensively implemented, assembly-based approaches are more limited. Of the three structural variant approaches, whole genome de novo assembly has been implemented by tools such as cortex, SOAPsv, laSV, and AsmVar, breakpoint assembly by SVMerge, TIGRA, MetaSV, BreaKmer, SoftSV, and manta, but break-end assembly has found very limited usage. Targeted break-end assembly has been used by CREST to assemble consensus sequences of soft-clipped read clusters, and NovelSeq combined de novo assembly with targeted break-end assembly of one-end anchored read clusters to identify large novel insertions but, to date, break-end assembly has been used only for these specialised purposes.

Conceptually, break-end assembly lies between de novo assembly and breakpoint assembly. When assembling, de novo assembly is performed reference free, break-end assembly is performed based purely on read mapping location, and breakpoint assembly is performed based on the read mapping location and prior identification of a putative structural variant. Uniquely, break-end assembly uses the prior information about the genome encoded in the reference genome alignment but does so in a manner which is not biased by the variant caller. Although in theory unbiased breakpoint assembly could be performed by considering all pairs of genomic locations, genome size makes this computationally infeasible. Thus breakpoint assemblers require prior identification of the candidate assembly location pairs. This can take the form of an explicit set of variant calls provided to the assembler (e.g. SVMerge, MetaSV, TIGRA, BreaKmer), or can be done internally from split read and/or
discordant read pair analysis (e.g. manta, SoftSV). Because of this, candidate assembly locations are biased towards loci that provide a sufficiently strong split read/discordant read pair signal.

Since break-end assembly of a given contig involves only a single genome locus, the loci identification stage required for breakpoint assembly is not necessary. Indeed, this single locus per contig property makes break-end assembly linear with respect to genome size thus genome-wide break-end assembly is computationally feasible.

A key limitation from the existing specialised break-end assemblers is their assembly of only a subset of the reads. A given genomic rearrangement could be supported by soft-clipped reads, split read, discordant read pairs, and one-end anchored read pairs which, taken together, allow for the assembly of a break-end contig. By treating this contig in the same way a split read is identified from a soft-clipped read, the partner break-end of the underlying breakpoint can be identified. Critically, the sequence being aligned can be up to the length of the library fragment size. This has important implications for the detection of rearrangements in repetitive sequence.

From a theoretical perspective, break-end assembly can detect a class of rearrangements not detectable by existing approaches. Were a breakpoint to occur between uniquely mappable sequence and repetitive sequence in which every originating read could be mapped to thousands of locations, this event would be able to be detected by break-end assembly if the resultant break-end contig was uniquely mappable. Such an event would have no split reads and no discordant read pairs supporting the event. Although a multi-mapping aware caller could in theory detect such an event, computation constraints constrain these callers and limit the number of positions that are considered for each read - a limitation not present when performing break-end assembly. How prevalent such events are in actual data remains to be seen, but GRIDSS’ correct assembly of SINE element breakpoint supported by zero split reads and zero discordant reads indicates that such events are not purely theoretical.
2.2 GRIDSS: sensitive and specific genomic rearrangement detection using positional de Bruijn graph assembly

Daniel L Cameron¹², Jan Schroeder¹²³, Jocelyn Sietsma Penington¹, Hongdo Do⁴⁵⁶, Ramyar Molania⁴⁷, Alexander Dobrovic⁴⁵⁶, Terence P Speed¹⁸, Anthony T Papenfuss¹²⁸⁹¹⁰

¹Bioinformatics Division, Walter and Eliza Hall Institute of Medical Research, Parkville, Victoria, 3052, Australia;
²Department of Medical Biology, University of Melbourne, Parkville, Victoria, 3010, Australia;
³Department of Computing and Information Systems, The University of Melbourne, Melbourne, Australia;
⁴Translational Genomics and Epigenomics Laboratory, Olivia Newton-John Cancer Research Institute, Heidelberg, Victoria, 3084, Australia;
⁵Department of Pathology, University of Melbourne, Parkville, Victoria, 3010, Australia;
⁶School of Cancer Medicine, La Trobe University, Bundoora, Victoria, 3084, Australia;
⁷Department of Medicine, University of Melbourne, Austin Health, Heidelberg, Victoria, 3084, Australia;
⁸Department of Mathematics and Statistics, University of Melbourne, VIC 3010, Australia.
⁹Peter MacCallum Cancer Centre, Victorian Comprehensive Cancer Centre, Melbourne, 3000, Australia;
¹⁰Sir Peter MacCallum Department of Oncology, University of Melbourne, Parkville, Victoria, 3010, Australia.

2.2.1 Abstract

The identification of genomic rearrangements, particularly in cancers, with high sensitivity and specificity using massively parallel sequencing remains a major challenge. Here, we describe the Genome Rearrangement IDentification Software Suite (GRIDSS), a high-speed structural variant (SV) caller that performs efficient genome-wide break-end assembly prior to variant calling using a novel positional de Bruijn graph assembler. By combining assembly, split read and read pair evidence using a probabilistic scoring, GRIDSS achieves high sensitivity and specificity on simulated, cell line and patient tumour data, recently winning SV sub-challenge #5 of the ICGC-TCGA DREAM Somatic Mutation Calling Challenge. On human cell line data, GRIDSS halves the false discovery rate compared to other recent methods. GRIDSS identifies non-template sequence insertions, micro-homologies and large imperfect homologies, and supports multi-sample analysis and optionally supports analysis of multi-mapping read alignments—further increasing sensitivity and improving the ability of GRIDSS to resolve rearrangements in repetitive regions of the genome. GRIDSS is freely available at https://github.com/PapenfussLab/gridss.
2.2.2 Introduction

Structural variants (SVs), including large deletions and chromosomal translocations, play a significant role in the development of cancer and other diseases. Historically, early links between SVs and cancer were found by cytogenetics [e.g. 127], including the discovery of circular chromosomes [128] and the Philadelphia chromosome [129, 130]. It was only with the advent of high-throughput sequencing that the extent of genomic rearrangement in cancer, the role of catastrophic chromosome-scale events such as chromothripsis [131], the broader clinical relevance of SVs [132] and their predictive value [133] were identified. Although significant progress has been made, SVs remain less well studied than single nucleotide variation, in part due to challenges in their reliable identification from short-read sequencing data.

Many methods exist to identify SVs using high-throughput sequencing data. These use one or more of three forms of evidence: read depth, split reads, and discordantly-aligned read pairs. Changes in read depth (RD) are associated with copy number variants and imply genomic rearrangements, but RD methods [e.g. 134, 135] cannot resolve genomic fusion partners and breakpoint positions are imprecise. Rather, resolution is dependent on the overall sequencing depth and the selected window size used in the analysis. Using paired-end sequencing, clusters of discordantly-aligned read pairs (DPs)—i.e. read pairs that align to different chromosomes, with unexpected orientation, or separation—can be used to infer the presence of SVs. Since the SVs are assumed to occur in the unsequenced part of the DNA fragments, DP methods [e.g. 32, 136] do not identify exact breakpoint locations. Single nucleotide resolution of SVs is useful for predicting possible fusion gene products or the impact of a promoter translocation, identifying disrupted tumour suppressors, determining the DNA repair mechanism responsible for the SV, and investigating motifs associated with the breakpoint. It is obtained using split reads (SR), where the sequenced reads span the breakpoint. SR methods [e.g. 37, 45, 46] find SVs by identifying split alignments in which part of the read aligns to either side of a genomic rearrangement, either through direct split read mapping by read aligner, realignment of soft clipped (SC) bases (unaligned bases in partially mapped reads), or split alignment of the unmapped read in one-ended anchored (OEA) read pairs (read pairs with only one read mapped) [137]. Whilst most tools consider only the best alignment for each read, some methods [32, 66] consider multiple potential alignment locations to better handle repetitive sequence [82]. Multiple approaches have been developed that incorporate two (e.g. DELLY [63]) or three (e.g. LUMPY [69]) forms of evidence in the variant calling process. Long reads are now also being used [138] or incorporated as evidence (e.g. Parliament [71]).
To improve SV calls, short read assembly has also been incorporated into methods in a variety of ways. Assembly of reads obtained from clusters of SC reads (e.g. CREST [45]) or Open End Anchors (OEs) (e.g. NovelSeq [62]) has been used to form break-end contigs, which extend out and span the breakpoint from each side. In contrast, breakpoint contigs are generated by local assembly of all reads supporting a rearrangement, generating a single contig supporting the variant. Some methods apply targeted assembly to validate the breakpoint calls (e.g. Manta [64], SVMerge [57], TIGRA [59]). Windowed breakpoint assembly has been used (e.g. SOAPindel [19], DISCOVAR [60]), but detection is limited to events smaller than the window size. Whole genome de novo assembly has also been used for variant calling (e.g. cortex_var [51]), but its use has been limited, in part due to the computational expense compared to alignment-based approaches.

Here, we present a novel approach to predicting genomic rearrangements from DNA sequencing data, GRIDSS (Genome Rearrangement IDentification Software Suite), which provides high-speed variant calling from a combination of assembly, split read and read pair support. The philosophy underpinning GRIDSS is to maximise sensitivity and prioritize calls into high or low confidence, thereby maintaining specificity in the high confidence call set. To achieve this, we take a three-step approach. Firstly, we filter out reads that align properly; i.e. we extract all reads that might provide any evidence for underlying genomic rearrangements. Secondly, we perform assembly of all remaining reads using a novel algorithm that utilises information from the alignment to constrain the assembly. We term this genome-wide break-end assembly, as each contig corresponds to a break-end and only after assembly is the underlying breakpoint and partner break-end identified. Unlike existing break-end assemblers that perform targeted assembly of soft clipped (SC) or one-end anchored (OEA) reads, our approach performs genome-wide assembly of all SC, SR, DP, OEA and indel-containing reads. Similar to split read identification from soft clipped reads, breakpoints are identified by realignment of break-end contigs. Finally, we apply a probabilistic model that combines break-end contigs from each side of the rearrangement with SR and DP evidence to score and call variants. GRIDSS can also optionally be applied to read alignments with multiple reported mapping positions.

To perform the genome-wide break-end assembly, we developed a novel assembly approach specifically for the task by extending a positional de Bruijn graph data structure. Originally developed for small indel and base calling error correction of de novo assembly contigs [139], positional de Bruijn graphs add positional information to each node, transforming them into a directed acyclic graph and making use of valuable information generated by the aligner. With appropriate optimisation, this is computationally efficient at the genome scale, and reduces depth of coverage needed and memory requirements for accurate assembly. To make the best use of data from related samples, sequencing
libraries are tracked in the de Bruijn graph using color [51], and evidence supporting rearrangements is shared between libraries during assembly and variant calling. Since the assembled contigs are longer than the read length, this improves performance in regions of poor mappability. Meaningfully scored variants and a set of useful default filters makes GRIDSS easy to use, but also a powerful tool for advanced users, who, armed with prior knowledge about expected rearrangements, can identify relevant calls with low support.

Using simulated and cell line data, we show that GRIDSS outperforms other methods. We demonstrate its effectiveness on cell line and patient tumour sequence data, including rearrangements with extreme complexity, as well as sequencing data from the AT-rich *Plasmodium falciparum* genome.

2.2.3 Results

2.2.3.1 Outline of GRIDSS

The GRIDSS pipeline comprises three distinct stages: extraction, assembly, and variant calling (Figure 2-1).
a) Soft clipped and indel-containing reads as well as discordant and one-ended anchored read pairs are extracted from input BAM files. Split reads are identified through re-alignment of soft clipped read bases. b) Extracted reads are added to a positional de Bruijn graph in all positions consistent with an anchoring alignment. Break-end contigs are identified by iterative identification of the highest weighted unanchored graph path followed by removal of supporting reads. Unanchored contig bases are aligned to the reference genome to identify all breakpoints spanned by the assembly. c) Variants are called
from assembly, split read, and read pair evidence using a probabilistic model to score and prioritise variants.

In the extraction stage, GRIDSS takes as input any number of SAM/BAM alignment files and calculates a number of summary statistics for each sequencing library. SRs are identified by re-alignment of the unaligned 3’ or 5’ “soft clipped” read bases back to the reference genome. DPs consist of read pairs aligned with inferred fragment size shorter or longer than 99.5% of read pairs, in the wrong orientation, or to different chromosomes. All SC, DP, OEA and indel-containing reads are then extracted. At this stage no SV calling has been undertaken. Reads are scored according to likelihood of originating from the reference allele based on the read mapping quality, the empirical distributions of the read alignment and library fragment size, and read mapping rate. Extracted reads are passed to the assembly stage, with split reads and discordant read pairs also passed to the variant calling stage.

In the assembly stage, reads are decomposed into a sequence kmer of k consecutive bases. These kmers and their genomic locations are incorporated into a positional de Bruijn graph. Kmers are considered anchored if the originating read aligns to the reference for all k bases and these anchoring kmers are used to constrain the positions in which a read can be assembled. Unanchored kmers from soft clipped reads are placed as if the read were fully mapped. Split and indel-containing reads are treated as two independent soft clipped reads—one for each alignment location. OEA read kmers are placed at all positions compatible with the alignment of the mapped read and the DNA fragment size distribution, with DPs treated as two independent OEA read pairs. Each read kmer is weighted according to the constituent base quality scores and variant support score, with graph nodes weighted by the cumulative supporting weights. Error correction is performed to remove spurious paths caused by sequencing errors. Break-end contigs are iteratively identified by finding the highest weighted unanchored path and extending into anchoring kmers if present. To ensure each read supports only a single contig, reads supporting each break-end contig are removed from the graph when the contig is assembled. Unaligned contig bases are iteratively aligned to the reference genome to identify all genomic rearrangements spanned by the assembly. Data streaming and graph compression is used extensively to keep the assembly memory footprint below 2GB per thread.

Variant calling occurs in the final stage of GRIDSS. Variants are identified from the overlap in predicted breakpoint positions of assemblies, SRs and DPs. After identifying and scoring all overlapping support sets, each SR, DP and assembly is then assigned to the highest scoring variant it supports. High-scoring variants with assembly support from both break-ends are considered high confidence calls. Variants that are low-scoring or lack assembly support from one or both break-ends are considered low confidence and a filter flag is applied to these variants in the output VCF. Non-template sequence
insertions as well as exact micro-homologies and large imperfect homologies are automatically identified in the variant calls.

2.2.3.2 Performance on simulated data

To assess the performance of GRIDSS, we simulated heterozygous structural variants with a range of event types (deletion, insertion, inversion, tandem duplication, translocation) and sizes (1 base pair (bp) to 65 kilo-base pairs) on human chromosome 12 (hg19). We compared the GRIDSS results to a number of other tools (BreakDancer, Pindel, DELLY, HYDRA_MULTI, LUMPY, SOCRATES, cortex, and TIGRA; see 2.3 Supplementary Material for full details).

For parameters typical of tumour genome sequencing (60x coverage of 100bp paired end reads with a mean fragment size of 300bp), GRIDSS obtained near-perfect sensitivity across the widest range of event types and sizes (Figure 2-2), albeit with Pindel having greater sensitivity on small (<50bp) events, and only the de novo assembly-based caller cortex able to detect large insertions. For both random breakpoints and breakpoints in SINE/ALU elements, only the multi-mapping aware caller HYDRA obtained sensitivity higher than GRIDSS (99.58/99.57% vs 99.36/99.22%), with DELLY, Socrates and LUMPY exceeding 95% sensitivity in both data sets (99.08/98.76%, 98.94/98.96%, 97.43/97.78%). BreakDancer, DELLY, Pindel, and TIGRA all incorrectly classified breakpoint events as inversion events. The assembly-based callers cortex, Manta and GRIDSS identified breakpoints with the greatest accuracy, followed by Pindel, SOCRATES, and DELLY (all of which incorporate split reads) (Figure 2-21). The positional accuracy of LUMPY was comparable to the purely read pair based callers BreakDancer and HYDRA. Except for the detection of insertion events using cortex, GRIDSS obtained a higher precision than other callers with comparable sensitivity (Figure 2-7).
Figure 2-2 Variant caller performance on simulated heterozygous genomic rearrangements

Different classes of genomic rearrangement were randomly generated against human chr12 (hg19) and 60x coverage of 2x100bp sequencing data was simulated. (A) The sensitivity of each method (rows) for each event type (columns) is plotted against event size. (B) Receiver
Operating Characteristic (ROC) curves for all breakpoints (left) and breakpoints located in SINE/ALUs (right).

To explore the impact of read length (36-250bp), DNA fragment size (100-500bp), sequencing depth (4-100x) and aligner (bwa, novoalign, bowtie2), GRIDSS was applied to a comprehensive simulation (see 2.3 Supplementary Material for further details). For reads 50bp or longer, GRIDSS is able to reliably call and assemble heterozygous genomic fusions at 30x coverage regardless of aligner or fragment size, although some libraries (such as 100bp paired end reads with 500bp fragment size) require as little as 8x coverage. Overall, the F1-scores of GRIDSS show improved call quality for increasing read length, read depth, and fragment size (Figure 2-8). Whilst the precision of calls supported by single-sided or no assembly decreased with coverage as expected, precision of calls supported by reciprocal breakpoint assembly remained near 100% regardless of sequencing depth, read length, library fragment size, or aligner (Figure 2-10). This demonstrates that requiring reciprocal break-end assembly support is a simple yet surprisingly powerful false positive filter. Although frequently overlooked or uncontrolled in experimental designs, our simulation results reveal the significant impact library fragment size has on structural variant calling. Unlike single nucleotide and indel calling, which are relatively independent of library fragment size, the impact of library fragment size on structural variant calling can be the equivalent of up to a two-fold change in coverage.

2.2.3.3 Performance on cell line data

We next tested GRIDSS on several real sequencing data sets. First, GRIDSS was applied to short read sequencing data from the NA12878 cell line (50X coverage PCR-free 2x100bp Illumina Platinum Genomes; ENA accession: ERA172924), along with several other structural variant callers. Callers were evaluated against both curated validated call sets [69, 140], and against PacBio and Moleculo long reads [141]. As previously [140], only deletions longer than 50bp were considered. So as to not unfairly penalise imprecise callers such as BreakDancer, calls were considered true positives if the breakpoint position error was less than the library fragment size, and the event length differed by at most 50% from the validated call set. For the PacBio and Moleculo long reads, variant calls required at least 3 split reads (with each split alignment mapping at least 25% of the long read), or 7 reads containing a corresponding indel, to support the event. Ambiguously mapped long reads with aligner-reported mapping quality of 0 were ignored. ROC curves for other callers were obtained by varying the required number of supporting reads as reported by the caller.
Figure 2-3 Performance of SV callers on deletion detection in NA12878 at 50x coverage

Multiple variant calls were compared to both the Mills et al validation call set (a and b) and PacBio/Moleculo long read orthogonal validation data (c and d). Plots show the number of true positives versus false positives (a and c) and the precision versus true positives (b and d). PacBio/Moleculo validation required 3 split, or 7 spanning long reads supporting the call. For both validated call sets, GRIDSS generally exhibits considerably better performance characteristics than other callers (Figure 2-3). When compared to the Mills et al call set, GRIDSS was able to identify the first 1,000 true positives with a false discovery rate of 7% (3% using PacBio/Moleculo validation data), compared to the next closest method, LUMPY, at 11% (7%). When an aligner that reports
multiple alignment positions for each read is used, GRIDSS sensitivity is improved by 25%, but at cost of slightly reduced specificity (see 2.3 Supplementary Material for details).

To determine the relative contributions of split read, read pair, and assembly support, GRIDSS was run on read pair (DP) and split read (SR) subsets with and without assembly. As expected, using more of the available evidence results in better variant calls (Figure 2–4). Assembly improves variant calling for DP and DP+SR evidence, but does not improve SR alone as the assembly contig lengths are limited to the length of the longest soft clip.

![Figure 2-4 Breakdown of GRIDSS variant calling performance.](image)

Performance comparison of NA12878 deletion events using PacBio/Moleculo orthogonal validation data. Precision versus the number of true positives for different types of support (a) and for different kmer sizes (b).

Assembly of both split reads and read pairs improves both sensitivity and specificity to levels not achievable by either evidence source. Scoring only assembly-supported variants and varying the type of assembly and kmer size demonstrates that robust small kmer break-end assembly can be achieved with positional de Bruijn graph assembly, but not windowed de Bruijn assembly.

To understand the contribution of the positional de Bruijn graph assembly, we tested a windowed break-end assembler using a traditional de Bruijn graph (see 2.3 Supplementary Material) in the GRIDSS framework. By restricting variant scoring to consider only assembly-based support, we compared windowed and positional assembly across a range of kmer sizes against the PacBio/Moleculo orthogonal validation data set. Positional de Bruijn graph assembly exhibited
minimal drop in performance from kmers 31bp to 22bp, while windowed assembly performance was highly sensitive to kmer size, and sensitivity and specificity were well below those of the positional assembly for all kmers tested (Figure 2-4). The poor performance of windowed assembly was traced to the incorrect assembly of kmers occurring in multiple positions within the assembly window (Figure 2-15). As expected, this phenomenon was especially pronounced in simple repeats and regions of low complexity (Figure 2-12). As this mode of mis-assembly does not occur in positional de Bruijn graph assembly, GRIDSS is able to perform assembly with shorter kmers and therefore at lower coverage than either windowed or de novo assembly.

2.2.3.4 Application to complex genomic rearrangements

To evaluate the performance of GRIDSS on complex genomic rearrangements, SVs were predicted in three published cancer-associated neochromosome datasets [142] (see 2.3 Supplementary Material). Each neochromosome contains hundreds of genomic rearrangements identified from the integration of copy number and discordantly aligned read pairs, followed by extensive manual curation. A high concordance with the 1010 curated SVs was obtained with GRIDSS detecting 98% of the curated calls, 92% with high confidence. GRIDSS obtained a higher concordance with the previously published curated results than other tested methods (BreakDancer, Cortex, DELLY, LUMPY, HYDRA, Pindel and SOCRATES; see 2.3 Supplementary Material).

GRIDSS also refined the original call set in three ways. Firstly, GRIDSS calls were made to single nucleotide resolution. Secondly, GRIDSS was able to identify 414 additional high confidence breakpoints, the majority (66%) of these were missing from the curated call set because they were supported by fewer read pairs than the fixed threshold applied in the original analysis, or were within 1000bp of another SV (potentially an issue due to the use of DP evidence alone). Finally, in 5% (64) of the SVs, GRIDSS was able to refine events classified as simple genomic fusions between two locations (A and B) that were in fact compound genomic fusions (from locations A to C to B), where the fragment C was short. In these events, GRIDSS was able to assemble a breakpoint contig at A, fully spanning the C fragment, with the remainder of the contig unambiguously aligning to B. A further 31 compound genomic fusions were identified in which the spanned fragment could not be unambiguously placed. While pure split read methods should also detect these compound rearrangements, the order in which DP and other evidence is applied and how it is applied will impact whether other methods can detect these features. This further refines the picture of complex rearrangements in neochromosomes and provides single nucleotide resolution of DNA breaks.
2.2.3.5 Application to cancer samples

To demonstrate the clinical utility of the low false discovery rate of the GRIDSS high-confidence call set, GRIDSS was used to identify patient-specific DNA biomarkers from tumour biopsies for application to cell free DNA. Somatic rearrangements were predicted from 40x coverage sequencing data from primary lung cancers from two patients without matched germline data (see 2.3 Supplementary Material for details). Primers were designed for 8 highest scoring somatic candidate SVs (four from each patient). All 8 SVs were validated by real-time PCR (only 6 were somatic, the remaining 2 were found to be real germline SVs).

Next, GRIDSS was compared to results from two previously published cancer data sets. First, SVs were identified from sequencing data from a melanoma metastasis and matched germline samples (60X coverage tumour; 30X coverage normal) [46]. GRIDSS detected 1,051,017 high and low confidence SVs with any level of support. Of these, 492 were high confidence calls and 851,981 had very low support (at most 3 supporting reads/read pairs). Of the 8 somatic events previously predicted by Socrates and validated by PCR [46], all 8 (100%) were identified by GRIDSS with high confidence.

Finally, GRIDSS was run on DNA-Seq data from the HCC1395 breast cancer cell line and results compared to the published genomic breakpoints, which were predicted to be associated with “validated” fusion genes [108]. GRIDSS showed strong concordance with the published results (Appendix A). Using a 21x coverage subset of the HCC1395 WGS data (only 21x of the 63x is publically available), GRIDSS identified 23 of the 26 published genomic breakpoints (22 with high confidence, 1 with low confidence), and failed to identify 1 breakpoint (manual inspection showed no supporting reads were present in the available data). The remaining 2 calls not identified by GRIDSS appear to be false positives caused by read-through transcription (see 2.3 Supplementary Material for details). Additionally, GRIDSS identified genomic breakpoints associated with a further 3 of the validated fusion genes.

2.2.3.6 Application to Plasmodium falciparum genome data

To test the behaviour of GRIDSS on a challenging AT-rich non-mammalian genome, it was applied to a laboratory strain of Plasmodium falciparum that was genetically modified for use as a live, attenuated malaria vaccine (C5) and to the parental 3D7 population (ENA accession: PRJE12838). In the vaccine candidate, a plasmodium gene KAHRP was knocked out by insertion of a construct containing a processed copy of the human DHFR transcript. In addition to identifying the insertion of the construct into the KAHRP locus, GRIDSS detected 8 of the 9 exon splicing events associated with the processed DHFR transcript with high confidence, while the ninth splicing event was found in the low confidence call set. This demonstrates the value of GRIDSS’ sensitivity and prioritization of calls. Finally, a tandem
duplication was also identified by GRIDSS in one of the VAR gene regions and supported by a copy number change (Figure 2-5). VAR genes are a large and complex gene family and the VAR gene regions are prone to recombination. The duplication was clonal in the C5 candidate and present at low frequency in the parental population (supported by 1 read), which was utilized in the positional de Bruijn graph to identify the rearrangement.

![Tandem duplication identified in a Plasmodium falciparum VAR-gene region](image)

Figure 2-5 A tandem duplication identified in a *Plasmodium falciparum* VAR-gene region

Coverage is shown for two samples of *P. falciparum*—a genetically modified line (C5), which was derived from the parental laboratory strain (3D7). The AT-rich genome shows high coverage in genes, which drops to very low levels in the AT-rich non-exonic regions. A change in copy number is apparent in the C5 coverage. GRIDSS detected the underlying tandem duplication in the C5 vaccine candidate (indicated). The supporting discordant read pair (DP) evidence is shown for both strains. Weak evidence (1 read pair) for this rearrangement was also detected in the parental population indicating that the SV was subclonal in this population. This evidence contributed to the coloured, positional de Bruijn graph assembly.

### 2.2.3.7 Runtime performance

Runtime performance of variant callers was compared for 50x coverage whole genome human sequencing, and variant-dense simulated data sets on a virtual machine with 20 cores on a dual-socket Xeon E5-2690v3 server. GRIDSS runtime is comparable to that of LUMPY, lags behind that of BreakDancer, SOCRATES and HYDRA, and is considerably faster than DELLY, Pindel, Cortex and TIGRA.
(Table 2). Of the 236 minutes wall clock time taken by GRIDSS, 159 minutes was spent on file I/O and parsing of the input over three separate passes (library metrics calculation, evidence extract, reference coverage annotation), with only 38, 11, and 11 minutes spent on assembly, soft clip alignment by external aligner, and variant calling respectively (Figure 2-20).
2.2.4 Discussion

We have developed the GRIDSS software package, which performs genome-wide break-end assembly, and combines assembly, split read, and discordant read pair evidence using a probabilistic model to identify and score structural variants. It automatically detects non-template sequence insertions, and both exact micro-homologies and large imperfect homologies. Through comparison with existing split read, read pair, assembly-as-validation, and de novo assembly approaches on both simulated and real data, we have shown that our approach significantly improves both sensitivity and specificity of variant calling. We also demonstrated that GRIDSS is effective on real data from human tumours as well as non-mammalian organisms.

GRIDSS is designed to make the most of available evidence. The break-end assembly algorithm is able to make use of SR, DP, OEA, SC and indel-containing reads, and utilizes information generated by the aligner via a positional de Bruijn graph. By combining SR, DP and assemblies into a unified probabilistic model, GRIDSS is able to call variants even in the absence of any two of these signals. Such an approach is a distinct advantage over callers such as DELLY that require a threshold signal strength in one signal before considering any others.

GRIDSS is designed to be highly sensitive and not miss any putative genomic rearrangements. GRIDSS uses variant score and the presence of assembly support from both sides of the rearrangement to classify variants as high or low confidence. The high confidence set provides an immediately usable call set with high specificity, which is particularly important in clinical applications. The retention of low confidence calls, as well as the option to call variants from reads with multiple reported alignments enables analysis requiring high sensitivity.

<table>
<thead>
<tr>
<th></th>
<th>SOCRATES</th>
<th>BreakDancer</th>
<th>HYDRA</th>
<th>LUMPY</th>
<th>GRIDSS</th>
<th>DELLY</th>
<th>Pindel</th>
<th>Cortex</th>
<th>TIGRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>NA12878 50x</td>
<td>52</td>
<td>82</td>
<td>85</td>
<td>211</td>
<td>236</td>
<td>489</td>
<td>2184</td>
<td>8174</td>
<td>13196</td>
</tr>
<tr>
<td>Simulation</td>
<td>88</td>
<td>74</td>
<td>147</td>
<td>258</td>
<td>440</td>
<td>702</td>
<td>2475</td>
<td>17460</td>
<td>3660</td>
</tr>
</tbody>
</table>

Table 2 SV caller runtime performance

Times shown are execution time in minutes. Benchmarking was performed on a dual socket Xeon E5-2690v3 server.
Our novel genome-wide break-end assembly approach is made possible through the utilisation of a positional de Bruijn graph to incorporate read alignment constraints into the graph structure itself. This allows us to perform genome-wide assembly without targeting or windowing, and use a kmer size half that required for de novo de Bruijn graph assembly. This small kmer in turn results in improved assembly at lower levels of coverage. Although our approach results in a graph two orders of magnitude larger than the equivalent de novo assembly de Bruijn graph, we are able to perform genome-wide break-end assembly faster than a number of existing targeted breakpoint assembly implementations through graph compression, data streaming, multi-threading, reference-supporting read exclusion, and extensive use of dynamic programming and memoization. By demonstrating that GRIDSS performance is comparable to existing callers when only discordant read pairs or split reads are considered, we show that the incorporation of positional de Bruijn graph-based whole genome break-end assembly into the variant calling process is key to the superior performance of GRIDSS.

GRIDSS has been released as free and open source software under a GNU General Public License (GPL version 3) and is available at https://github.com/PapenfussLab/gridss.

2.2.5 Methods
GRIDSS has been designed for 36-300bp Illumina paired-end or single-end DNA-Seq sequencing data, and accepts any number of coordinate sorted SAM, BAM or CRAM input files, with no requirement for matching read lengths between input files. Mate-pair libraries are currently not supported. User-supplied categories for each input file allow for somatic and multi-sample calling with support and variant scoring broken down per category. Variant calls are output to VCF using the break-end notation of VCFv4.2.

2.2.5.1 Identification of supporting reads
An initial pass of the read alignments is performed to collect library metrics. Metrics are calculated independently for each input file, thus allowing libraries from multiple related samples (or unrelated samples from a population) to be processed together. The following metrics are gathered in the initial pass for each input file: read alignment CIGAR element length distribution, fragment size distribution, and counts of total reads, mapped reads, unmapped reads, total read pairs, read pairs with both reads unmapped, read pairs with one read unmapped, read pairs with both read mapped, maximum read length and maximum mapped read length. The fragment size distribution is calculated using Picard tools CollectInsertSizeMetrics.

Reads partially aligning to the reference (reads containing indels or soft clipped bases) and read pairs in which the inferred fragment size falls outside the expected fragment size distribution (default
shorter or longer than 99.5% of read pairs) of the library, with incorrect orientation, or with only one
read mapped are extracted to intermediate files. For multi-mapped input data, read pairs are
considered concordantly mapped and not extracted if the inferred fragment size of any pairing of
mapping locations falls inside the expected fragment size distribution. GRIDSS does not utilise read
pairs in which both reads are unmapped.

2.2.5.2 Split read identification
Split reads are identified by aligning the soft clipped bases of partially aligned reads back to the
reference using bwa (default) or another aligner that is compliant with the SAM file format
specifications (such as bowtie2). Except for the case where multi-mapping alignment is used, reads
for which the soft clipped bases are uniquely aligned to the reference (default MAPQ ≥10) are
considered to provide split read support. Reads containing insertions or deletions in the read
alignment are treated as split reads aligning to either side of the indel.

2.2.5.3 Read filtering
Due to the prevalence of artefacts in sequencing data, GRIDSS includes a number of default filters
designed to improve specificity by removing reads likely to interfere with variant calling. By default,
the following read filters are applied:

- MAPQ of at least 10. The read mapping quality score (MAPQ) is defined by the SAM
  specifications as $-10 \log_{10}(\Pr(\text{incorrect mapping}))$. A low mapping quality score indicates
  that the read is unlikely to be mapped correctly. All reads with a MAPQ score of less than 10
  are treated as unmapped. For split reads, both the read MAPQ and the soft clip re-alignment
  MAPQ must exceed the threshold. This threshold can be adjusted and is not applied during
  the analysis of multi-mapping alignments.

- Maximum coverage of 10,000x. Regions of extremely high coverage may be due to a collapsed
  reference sequence in highly repetitive regions. Coverage of canonical sequences such as
  those present in the centromeres can be greater than 3 orders of magnitude higher than the
  nominal coverage and as such, regions with coverage greater than 10,000x are excluded from
  variant calling and assembly.

- Soft clipped reads must have at least 95% of aligned bases matching the reference sequence
  as these partially aligned, poorly matching reads are frequently alignment artefacts.

- Soft clipped reads must contain at least 4 soft clipped bases. Sequencing errors in the bases
  near either end of a read will result in Smith-Waterman local alignment of the read excluding
  the end bases from the alignment.
• Shannon entropy of aligned bases must be at least 0.5 bits. Reads with low complexity in the mapped bases (such as poly-A runs) were found to have a high false positive rate. If the Shannon entropy of the unclipped bases of a read is less than 0.5 bits, the read is considered unmapped. This threshold is not applied during the analysis of multi-mapping alignments.

• Mean Phred-scaled base quality score of soft clipped bases must be at least 5. Soft clips containing runs of low base quality scores are filtered as such sequences are typically due to sequencing errors.

• Adapter filtering. Reads containing the first 12 bases of the Illumina Universal Adapter, Illumina Small RNA Adapter, or Nextera Transposase sequences are filtered if the sequence is homologous to the reference sequence at the aligned location.

• Overlapping read pairs from short fragments are excluded. A read pair aligned in the expected orientation with overlapping read alignments is indicative of pairs sequenced from a fragment shorter the combined read lengths. Such read pairs do not provide support for structural variants and are filtered. If soft clipped, each read is still individually considered for split read analysis and assembly. In the case that the fragment length is shorter than the read length, the reads may contain soft clips ‘dovetailing’ in adapter sequence past the other read. If there exists a micro-homology between the adapter and reference adjacent to the alignment position, the reads will be over-aligned and such read pairs will appear to be in an unexpected orientation. Such edge cases are filtered by allowing a 4bp dovetail when determining whether a read originates from a short fragment.

2.2.5.4 Positional de Bruijn graph assembly

A positional de Bruijn graph is a graph where each node contains a sequence of k bases and the position at which those k bases are expected to occur. The positional de Bruijn graph \( G = (V, E) \) consists of the vertex set \( V \) and the edge set \( E \subseteq V \times V \). Let \( n = (S_n, p_n) \in V \) where \( S_n = (b_1, b_2, \ldots, b_k) \) is a kmer of \( k \) bases, and \( p_n \in \mathbb{Z} \) is the expected genomic position of the kmer. By definition, nodes are connected if they have adjacent kmers and positions:

\[
(n_1, n_2) \in E \iff p_1 + 1 = p_2 \land \forall_{i=1}^{i=k-1} b_{i+1} = b_{i+2}
\]

Unlike a traditional de Bruijn graph, \( G \) is a directed acyclic graph since \( \forall_{i,j} (n_i, n_j) \in E \rightarrow p_i < p_j \); any cycle would contain an edge in which \( p_i \geq p_j \). That is, all paths through the graph are simple (not self-intersecting) paths since traversal of every edge must advance the position by a single base. This allows algorithms such as the longest path problem that would require exponential time on a de Bruijn graph to be completed in polynomial time when applied to a positional de Bruijn graph.

In addition to the kmer and position, the following node attributes are used:
• \(w(n)\) is the node weight corresponding to the Phred-scaled probability of the supporting read kmer. The probability of the \(i\)th kmer of read \(r\) is given as the joint probability of every kmer base being correctly called by the sequencer, and the aligner mapping the read to the correct location. That is,

\[
w(r, i) = -10 \log_{10} \left( 1 - \left( 1 - 10^{-\text{mapq}(r)} \right) \cdot \prod_{i=1}^{k} \left( 1 - 10^{-\text{baseq}(r)} \right) \right)
\]

where \(\text{baseq}(r)\) is the Phred-scale base quality score of the \(i\)th base, and \(\text{mapq}(r)\) is the mapping quality score of read \(r\). As \(w(r, i)\) is Phred-scaled, \(w(n) = \sum w(r, i)\).

• \(\text{anchored}(n)\) is a boolean flag indicating whether the node bases are fully aligned to the reference genome in any supporting read. Note that this definition does not require the read bases to actually match the reference and both matched and mismatched base alignments are considered to be anchored if aligned.

• \(\text{support}(n)\) is the set of reads providing support for the given kmer at the given position.

2.2.5.5 Assembly graph construction

All extracted reads are added to one of two positional de Bruijn graphs based the expected break-end orientation. Reads supporting a rearrangement after the anchored position are added to the forward graph, whereas reads supporting a rearrangement before the anchored positions are added to the backward graph. Each read kmer is added to the graph at all expected mapping positions. For a soft clipped read, each read kmer is added to \(G\) at the position the kmer would start at if the entire read were mapped to the reference. Split reads are treated as two independent soft clipped reads. For read pairs, read kmers are added at all positions in which the read pair would be considered concordantly mapped based on the mapping location of the partner. For discordant read pairs, each read is added based on the anchoring location of the mate irrespective of the actual mapping location of the read, whereas one-end anchors are only added at one location as the unmapped read can provide no positional constraints on its partner.

Using this method of graph construction, correctly assembled unanchored paths are limited in length to less than the read length if supported by only soft clipped or split read evidence, and by the maximum concordant fragment size if supported by read pair evidence.
2.2.5.6 Assembly graph error correction

Base calling error correction is performed by collapsing similar paths. Paths are scored according to the sum of the node weights. A path A is collapsed into an alternate path B if both the total path weight of A is less than B, A and B differ by less than a fixed number of bases (default 2), and either both paths are same length and share the same start and end node (bubble popping), or A shares a start or end node with B and contains a single terminal leaf node (leaf collapse). By default, error correction is only performed on paths of length less than twice the read length that are either simple bubbles or terminal leaves.

For each node, all leaf and branch paths under the maximum length originating from the node are identified by traversal of branchless descendants. For each path identified, breadth first graph traversal is performed to identify candidate paths to merge. Memoization is used to track the optimal paths to each node thus reducing worst-case traversal complexity from exponential time to quadratic.

2.2.5.7 Break-end contig assembly

A break-end contig path consists of a sequence of adjacent unanchored nodes, optionally flanked by a sequence of anchored nodes. Paths flanked by anchoring nodes are called before unflanked paths. Maximally weighted paths are calculated in the same manner as error correction traversal with breadth-first traversal with memoization of the highest-weighted partial path at each node.

Once the maximally weighted path has been determined, the path is extended into flanking anchored nodes until the anchored path length exceeds both the maximum read length and the unanchored path length. Once the maximally weighted path contig is called, all reads supporting any unanchored kmer on the contig path are removed from the graph. The removal of supporting reads from all graph nodes ensures that each read contributes to a single assembly only. Contigs are iteratively called until no non-reference kmers remain in the graph.

Assembly contigs supported by less than the minimum required support (by default 3 reads) and unanchored contigs shorter than the read length are filtered.

2.2.5.8 Contig error correction

While positional information at nodes significantly reduces the rate of mis-assembly compared to windowed assembly, branch traversal introduces new modes of mis-assembly. When a read pair is self-intersecting or contains repeated kmers, the resultant contig will loop for as long as the fragment size window will allow. This mis-assembly can occur even with single read. For example, with k=4, the single read TAAAAC expected to start at one of the positions in the interval [10, 15], will result in the highest weighted path of TAAAAAAAAC starting at position 10. To prevent such mis-assemblies, kmer
chaining of the supporting reads is used to truncate the called path at the first kmer transition not supported by any constituent read. Truncation is performed starting from both the start and the end of the contig with the highest weighted truncated path called, preferentially calling anchored paths. Each supporting read is aligned to the contig position with the greatest number of matching kmers (breaking ties towards the truncation start kmer) and all kmer transitions supporting by the read are marked. Once all supporting reads have processed, the contig is truncated at the first kmer transition not supported by any reads.

2.2.5.9 Analysis of multi-mapping alignments

For assembly purposes, multi-mapped reads are assigned a mapping location of the highest weighted assembly contig containing that read. After all assemblies have been generated, a lookup table containing the read alignment that contributes to the highest scoring assembly for that read/read pair is generated. The contributions of multi-mapping reads in sub-optimal alignments are then removed from the associated assemblies. The memory usage of this lookup is limited to 56 byte per multi-mapped read through the use of an off-heap memory store and by using the upper 96 bits of the Murmur128 hashes of strings encoding the read names and alignments as keys.

2.2.5.10 Contig re-alignment

Once an assembly contig has been called, a multi-stage re-alignment process is used to identify the breakpoint supported by the contig. Assemblies containing at least one anchored base undergo Smith-Waterman (local) re-alignment around the expected contig position. Assemblies that fully align to the reference are treated as a fully-aligned indel-spanning assembly if an indel is present in the alignment, or filtered out as a false positive if the full alignment contains no indels. For unanchored assemblies with no soft clip or split read support, a breakpoint position interval is calculated based on the breakpoint interval consistent with the greatest number of supporting read pairs.

The contig bases not anchored to the reference are aligned using an external aligner (by default bowtie2) in local alignment mode using the same alignment thresholds used for identifying split reads. For assembly alignment, if the external aligner identifies the best alignment to be a soft clipped alignment, these soft clipped bases are again aligned, with such recursive alignment limited to a depth of 4. This compound re-alignment results in assembly support not only for the breakpoint site identified by the initial re-alignment, but also for any additional breakpoints spanned by the assembly. This approach allows accurate classification of complex rearrangements such as those present in neochromosomes formed through chromothripsis and the breakage-fusion-bridge process.
2.2.5.1 Detection of micro-homology and non-template sequence insertions

Depending on the pathway involved in DNA repair resulting in a structural variant, there may exist either sequence homology at the breakpoints, or non-template sequence inserted during the repair. Micro-homology at the breakpoints introduces uncertainty into the breakpoint position call, which is resolved if non-template sequence is also present. As a consequence of aligner behaviour, unhandled sequence homology can result in two separate variant calls (one at each edge of the homology) for a single event as aligners are able to map read to the homologous sequence at both side of the breakpoint. Unhandled non-template sequence insertions results in incorrect breakpoint sequence and event size calculation and, for coding variants, result in incorrect gene fusion transcript prediction.

GRIDSS factors in both breakpoint sequence micro-homology and non-template sequence insertions when performing variant calling. For micro-homologies, the breakpoint location for split reads and assemblies is expanded from a single base to an interval of the length of sequence homology between the read/assembly and the reference sequence at either side of the predicted genomic rearrangement. The nominal position of the called breakpoint is considered to be the centre of the homology, and is reported using the standard HOMSEQ, HOMLEN, and CIPOS fields in the VCF output. Non-template sequence insertions are included as an additional component of the variant call in a similar fashion to SOCRATES [46].

In addition to the standard VCF homology fields, GRIDSS reports the inexact homology in the non-standard IHOMPOS field. The inexact homology length is calculated for each break-end by performing local Smith-Waterman alignment of the breakpoint sequence to the reference sequence up to 300bp either side of the break-end.

2.2.5.2 Graph compression

To reduce the graph size, a compressed representation of the positional de Bruijn graph is used. Nodes with a single successor connected to nodes with a single ancestor correspond to a branchless path and are compressed into a single path node. Similarly, nodes with adjacent positions and matching kmers, weights and reference status are compressed using a position interval. The resultant path nodes are of the form \((\text{start}, \text{end}, (kmer_1, \text{weight}_1, \ldots, kmer_n, \text{weight}_n), \text{anchored})\), representing the set of nodes \(\bigcup_{p=\text{start}}^{\text{end}} \bigcup_{i=1}^{n} (p, kmer_i, \text{weight}_i, \text{anchored})\).

2.2.5.3 Streaming assembly

Since all nodes are associated with a genomic position and only have edges to adjacent genomic positions, all graph operations only affect the local subgraph near the genomic position. Exploiting this, positional de Bruijn graph assembly is performed in a single streaming pass over the input reads.
Coordinate sorted records are streamed through the following 5 internal processes, each processing records within a genomic position window size determined by the maximum read length and fragment size:

1. **Extraction**: each read is converted into constituent positional de Bruijn graph nodes. To reduce the graph size, positional intervals are stored implicitly with each extracted node defined over a positional interval.

2. **Positional aggregation**: overlapping nodes from multiple reads are split into disjoint aggregated nodes and graph edges calculated and cached.

3. **Path compression**: non-branching aggregate node paths are compressed into path nodes.

4. **Error correction**: bubble popping and leaf collapse error correction is performed to remove base calling artefacts.

5. **Assembly**: maximal path contig calling is performed such that whenever a maximal path is encountered, the streamed assembly graph loads and traverses all alternate paths which any reads supporting the maximal path could also support. Since each read can contribute to graph node positions over an interval no wider than the concordant fragment size plus the read length, the globally maximal path containing any given read must overlap this interval and is thus a local graph operation. A contig is called whenever the subgraph has loaded all potential alternate paths for the highest weighted maximal path encountered. Since all reads contributing to the called path must have been fully loaded for all such potential alternate paths to have been traversed, all reads contributing to the contig are fully removed from the subgraph immediately. This approach ensures that the any contig called is the globally maximal contig containing the given reads.

This assembly algorithm is implemented by memoization of all maximal paths of the streaming subgraph. The starting position of paths for which a potential successor node has not yet been loaded are tracked in a frontier and when the end position of the maximal path plus the concordant fragment size is earlier than both the earliest start position of the frontier path and the start position of the next node to be loaded, the maximal contig is called.

As a full recalculation of all maximal paths in the subgraph after supporting reads have been deleted is unnecessary, only paths affected by node removal are recalculated. For sufficiently high-coverage data, there will be enough concordant fragments with unexpectedly long or short fragment size that a background signal supporting small indels everywhere across the entire genome will be present. When these reads are assembled, this signal results in long unanchored paths up to multiple megabases in size. To supress this background noise, contigs and paths longer than the maximum
expected size are filtered and thresholds are placed on both the size and the genomic width of the loaded subgraph (See 2.3 Supplementary Material).

2.2.5.14 Coverage blacklist
Alignment of whole genome sequencing data to the reference results in extreme coverage in some genomic regions, for example in satellite repeats. These regions are likely to contain false positives and are computationally expensive to assemble due to the high coverage. To mitigate the effect of such regions on variant calling and runtime performance, GRIDSS performs two independent operations. Firstly, GRIDSS filters all reads within 1 fragment length of any region of extreme coverage (default 10,000x). Secondly, GRIDSS down-samples before assembly in poorly mapped regions, where the density of variant-supporting reads is unusually high. When the number of SV-supporting reads within a sliding window (default 1,000bp) exceeds the threshold (default 5,000), only a subset of the reads are used for assembly. The probability of a SV-supporting read being retained is given by

\[ Pr(read) = \begin{cases} \frac{1}{e^{-\frac{size-a}{m}}}, & \text{size} \leq a \\ 1, & x \geq 0 \end{cases} \]

where \( m \) is the target maximum number of reads in the window, and \( a \) is the unconditional acceptable threshold in which no downsampling occurs (default 50% of maximum). This hybrid exponential back-off model ensures that no down-sampling is performed except for coverage outliers, and within coverage outliers, real events are unlikely to be filtered.

2.2.5.15 Probabilistic variant scoring model
To estimate the quality of predicted structural variants, we score variants according to the Phred-scaled probability of originating from the mapped locations without any underlying structural variations. The Phred score \( Q \) of a probability \( P \) is given by \( Q = -10 \log_{10}(P) \). Given a read pair or split read \( r \) mapping to genomic locations \( a \) and \( b \), and the event \( M \) that the mapping is correct, the score assigned to \( r \) is given by \( Pr(r) = Pr(r \cap M) + Pr(r \cap \neg M) = Pr(M) \cdot Pr(r|M) + Pr(\neg M) \cdot Pr(r|\neg M) \). The probability of correct mapping \( Pr(M) \) is determined directly from the Phred-scaled mapping quality scores \( mapq_a(r) \), \( mapq_b(r) \) defining the probability of incorrect read alignment:

\[ Pr(M) = (1 - 10^{-mapq_a(r)/10}) (1 - 10^{-mapq_b(r)/10}) = 1 - Pr(\neg M) \] since \( Pr(M) \) requires both mapping locations to be correct. In the case of incorrect mapping, \( r \) is uninformative and \( Pr(r|\neg M) = 1 \). In the case of correct mapping, \( Pr(r|M) \) is determined empirically from the relevant library distribution.

For split reads, correct mapping with no SV implies the alignment is artefactual. We model the artefactual alignment rate from the empirical soft clip length distribution of the library thus
\[ \Pr(r|M) = P_{sc}(l_{sc}(r)) \]

where \( l_{sc}(r) \) is the length of the soft clip of \( r \) before split read identification, and \( P_{sc} \) is the library soft clip length distribution.

For read pairs, correct mapping can be caused either by a chimeric fragment \( CF \), or an unexpectedly large/small originating fragment:

\[ \Pr(r|M) = \Pr(r \cap CF|M) + \Pr(r \cap \overline{CF}|M) = \Pr(CF|M) \cdot \Pr(r|M \cap CF) + \Pr(\overline{CF}|M) \cdot \Pr(r|M \cap \overline{CF}). \]

\( \Pr(r|M \cap CF) \) is given as \( P_{rp}(if \ s(r)) \) where \( if \ s(r) \) is the fragment size of \( r \) inferred from the read mapping locations, and \( P_{rp} \) is determined from the library fragment size distribution inferred from all mapped read pairs. Chimeric fragment alignments are considered uninformative and \( \Pr(CF|M) \) is taken to be \( p_{ul} \), the rate of read pair mapping in which the inferred fragment size exceeds more than 10 mean absolute deviations from the median library fragment size. Thus for read pairs \( \Pr(r|M) = p_{d} + (1 - p_{d}) \cdot P_{rp}(if \ s(r)). \)

Split reads originating from indels use the corresponding distribution of insertion or deletion alignment operations instead of the soft clip distribution. Assemblies are modelled as a set of constituent reads with the anchored mapping quality defined as the greatest mapping quality of the constituent reads, and unanchored mapping quality determined by the assembly alignment mapping quality. Constituent soft clipped reads and reads with unmapped mates are treated as split reads and discordant read pairs for the purpose of determining assembly quality with the caveat that reads with unmapped mates use \( p_{u} \) in place of \( p_{d} \). This assembly scoring model improves variant calling by rescuing poorly mapped reads, increasing the score of variants supported by assembly, and promoting SC and UM reads to SR and DP reads within the context of the assembly thus allowing these reads to be used as input to the variant calling. For computational efficiency, scoring calculations are approximated using the maximum Phred score of the constituent terms.

### 2.2.5.16 Variant calling using maximal cliques

Variants are scored according to the level of support provided by SR, DP and assembly evidence combined. Supporting evidence can be summarised as the tuple \( (s_i, e_i, s_h, e_h, d_i, d_h, w) \) where the intervals \([s_i, e_i]\) and \([s_h, e_h]\) are the genome intervals between which a breakpoint is supported; \( d_i \) and \( d_h \), the direction of the supported breakpoint; and \( w \) the weight of the evidence as defined by the evidence scoring model. Since each piece of supporting evidence is considered to be independent, and evidence scores are expressed as Phred scores, the score for any given variant is equal the sum of the scores of evidence supporting the variant breakpoint.

Calculating the total support weight for all putative breakpoints is equivalent to finding all maximum evidence cliques, that is, all sets of evidence providing consistent support for a breakpoint that no more evidence can be added and the set remain consistent. Since both direction \( d_i \) and \( d_h \) must match
if evidence is to mutually support a breakpoint, the evidence set can be divided into the four ++, +-, -+
++, -- directional subsets which reduces the problem to weighted maximum clique enumeration of a
rectangle graph [Lee 1983]. Maximal clique enumeration is performed in a single in-order pass over
evidence in polynomial time.

Unfortunately, this approach can result in reads providing support to multiple independent
breakpoints. Since each read will have originated from at most one of the competing explanatory
variants, a second pass is made and a greedy assignment is performed in which each piece of evidence
assigned to only the highest scoring variant it supports. For reads with multiple alignments reported,
the read is considered to be aligned to the location supporting the highest scoring variant.

2.2.5.17 Breakpoint position error margin
Split reads originating from the same event should only support a single breakpoint position. In
practice, however, sequencing errors, imperfect homologies in the neighbourhood of the breakpoint,
and alignment artefacts can result in variation of the breakpoint position supported by each read. As
a result, split reads originating from the same event can provide support for breakpoints at different
positions. To ensure such evidence results in a single call, an additional margin is added when
determining clique assignment. By default, this margin is 10bp for variants 20bp or larger, linearly
decreasing to no margin for 1bp thus ensuring that small non-overlapping indels are not merged.

2.2.5.18 Software development methodology
GRIDSS was developed as professional quality software using a test driven development methodology.
To develop new functionality, test cases were first written describing the expected behaviour under
normal and error conditions. Once such failing test cases have been written, code implementing the
feature is written thus ensuring that the feature is functioning as expected. Bug fixing is performed by
first creating a failing test case reproducing the error, then updating the implementation to correct
the error. As a result, an extensive test suite composing of over 1,200 test cases has been developed.
Git is used as a version control system. All tests are rerun prior to each release, ensuring regression
faults in existing functionality are not introduced when new features are added.

Bug fixes and enhancement in libraries used by GRIDSS have been contributed back to these upstream
libraries. Maven2 is used for build packaging and a single precompiled binary including all
dependencies (except the external aligner) is produced for each release. All parameters used by
GRIDSS (including the choice of external aligner used) have been externalised into a configuration file
able to be modified by advanced users. Semantic versioning is used for release versioning.
GRIDSS is implemented in Java 1.8. GRIDSS has been designed as a modular software suite. Although an all-in-one entry point is included, each stage of the GRIDSS pipeline, including the break-end assembler, can be run as an independent program, or replaced with an equivalent implementation. Example scripts for single sample, somatic, and multi-sample pipelines are provided. Java utility programs and R scripts to convert GRIDSS VCF break-end format to more user-friendly formats for downstream filtering and analysis are also provided. All scripts required for independent replication of results presented in this paper are included in the GRIDSS source code.

2.2.6 References
References are included in Chapter 6 Bibliography.

2.3 Supplementary Material

2.3.1 Simulated data
To determine the accuracy and sensitivity of GRIDSS, and to compare performance to other methods, we developed an extensive set of simulated data. Our approach was to simulate a range of events on human hg19. For performance reasons and due the inability of some variant callers to detect inter-chromosomal events, only intra-chromosomal events were simulated. Chromosome 12 was chosen as it has close to the median chromosome size and GC content of the human genome, happens to be rich in oncogenes, and has been previously used for similar simulations. To allow false positive calls to be assigned based on event type, each simulation consists of events of a single type only. Six heterozygous variants sets were created: deletion, insertion, inversion and tandem duplication, and two translocation sets. For each of the simple event data sets, events of size 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 16, 20, 24, 28, 32, 48, 64, 80, 96, 112, 128, 160, 192, 224, 256, 288, 320, 512, 1024, 2048, 4096, 8192, 16384, 32768, and 65536 base pairs (bp) were simulated for a total of 18,000 variants of each type. Variants were positioned such that each variant had at least 2,500bp separation from all ambiguous reference bases and all other variants.

To establish relative detection capabilities of the variant callers in complex genome rearrangement environments such as those occurring in chromothripsis, translocation events were simulated by fragmentation of hg19 chr12 into 2500bp segments and concatenation of a subset in random orientation.

Two rearrangement data sets were generated, one reflecting random rearrangement events, in which the fragmentation was performed at random, and a second designed to contain more difficult to detect events. This second set was generated ensuring each rearrangement contained repetitive sequence on one side by selecting only 2,500bp sequences starting or ending at a SINE/Alu element.
as annotated in the UCSC hg19 RepeatMasker annotation. In both data sets, fragments with ambiguous reference bases were excluded from reassembly. In both data set, the variant chromatin contains 10,000 fragments.

For each of the 6 data sets, paired-end reads were generated at 100x coverage using the supplied Illumina MiSeq error profiles of ART 1.51 with 100bp read length and 300bp fragment size with a standard deviation of 10% of the mean fragment length. Reads were aligned to the full hg19 reference genome using bwa mem 0.7.10-r789 and coverage was downsampled to 60x, 40x, 15x, 8x and 4x using Picard tools.

Software used for comparison was selected favouring popular tools with a cross-section of calling methods. Comparisons were made using the most recently released versions as of October 2015. GRIDSS results were compared to BreakDancer 1.4.5 (breakdancer-max), CREST 0.0.1, CLEVER 2.0rc3, CORTEX 1.0.5.21, DELLY 0.6.8, GASVPro 20140228, HYDRA-multi 0.5.2, LUMPY 0.2.11, Pindel 0.2.5b6, SOCRATES 1.1, TIGRA 0.3.7 (using TIGRA-ext on BreakDancer calls), and VariationHunter (CommonLawRelease 0.04 using mrfast 2.6.0.1).

All software was run according to the settings recommended by the user guide, software website, README, or publication in that order of preference. Conversion scripts to VCF version 4.1 were written for software using custom output formats with the VCF calling location set to the start of the confidence window and the CIEND field set to the end of the confidence window. Memory and CPU statistics were collected for each caller by invoking the POSIX time command against a wrapper bash shell script. Variants were categorised as true positives if all event breakpoint locations were within 100bp of the called interval and the called event size was within 25% of the actual event. As BreakDancer does not report event orientation, breakend orientation was not considered when determining a match for any caller. For the chromothripsis-like simulation data set, variants 50bp or smaller were filtered to remove artefacts caused by inexact sequence homology around the breakpoint positions. For performance reasons, callers other than GRIDSS were run only on the 2x100bp, 300bp fragment bwa data slice for a total of 439,540 simulated variants per caller.

Scripts used to perform the simulations and comparisons are located in the src/test/sim and src/test/r subdirectories of the GRIDSS source code.

A number of structural variant callers were excluded from analysis for the following reasons:

GASVPro: results were significantly worse than expected for a read pair-based caller. Upon investigation it was found that the LLR scoring used by GASV excessively favoured very large events. Combined with a filter removing overlapping events, most true positive GASV calls were filtered by
the cluster pruning algorithm. For example, in NA12878 a false positive deletion call from chr1:32,060,879 to chr1:243,114,737 with a LLR of 1.4x10^{10} resulted in the removal of all chr1 deletion calls within this interval, thus removing most chr1 calls. Under these circumstances, it was decided that GASVPro results were not representative of read pair-based caller performance and results were excluded.

CLEVER: on simulated data, CLEVER called many fewer results than expected (41 deletion calls, 0 for other events). Through personal correspondence with the software author it was determined that all published version (1.1, 2.0rc1 and 2.0rc3) contained critical bugs causing either program failure, or call failure). As the recommended solution to compare against the most recent unstable, unreleased development version lacking both version and release information is not appropriate for reproducible evaluation, CLEVER results was excluded.

CREST: CREST failed to complete on 5 data sets. CPU usage indicated that CREST was running, but the output showed no progress was being made. For one data set, CREST was left to execute for 120 days without progress before being forcibly terminated. Although CREST could be run on the failing data sets if aligned with bowtie2, this may have confounded CREST results. Thus CREST was excluded.

laSV: laSV could not be run as the required auxiliary reference genome files were not present in the http://hgdownload.soe.ucsc.edu/goldenPath/hg19 reference data set, and no documentation was provided regarding the extraction format for the custom RepeatMasker track used by laSV. Additionally, neither the pre-packaged reference nor the example test data for laSV were publically accessible as of 20 December 2015.

BreaKmer: not designed to scale for whole-genome analysis\(^1\).

VariationHunter: No results for VariationHunter could be obtained. VariationHunter crashed with a “Segmentation fault (core dumped)” error on all simulations. VariationHunter could not be run on NA12878 in a timely manner due to requirement of in excess of 20,000 hours of computation time required to run mrfast on the 50x whole genome sequencing data set.

\(^1\) https://github.com/ccgd-profile/BreaKmer/issues/5
Figure 2-6 Comparison of simple event calls by read depth

At low levels of coverage, GRIDSS is able to detect large events at lower levels of coverage than the other calls, but such events are not in the default GRIDSS call set due to the low levels of support. Non-linear scale has been used to allow differentiation of callers with high sensitivity.
GRIDSS maintains the highest sensitivity at all read depths but, as with simple events, low coverage events typically have one or more filters applied. Non-linear scales have been used.
2.3.2 Comprehensive simulation and analysis of translocations

To further test GRIDSS, a comprehensive simulation across a range of read lengths, fragment sizes and depths of coverage was performed on GRIDSS using the chromothripsis-like simulation. Using the same protocol as the earlier simulation, paired-end reads were generated at 100x coverage using ART 1.51 with read lengths of 36, 50, 75, 100, 150 and 250bp and normally distributed fragment lengths of 150, 200, 250, 300, 400 and 500bp with standard deviation of 10% of the mean fragment length. Reads were aligned using novoalign 3.02.13 and bowtie2 2.2.5 as well as bwa mem 0.7.10-r789 and coverage down-sampled to 60x, 40x, 15x, 8x and 4x.

GRIDSS was run on all combinations of read length, read depth, fragment size and aligner except combinations in which the mean fragment size was less than or equal to the read length. A total of 13,831,272 variants were simulated. Performance for each is summarised by taking the harmonic mean of sensitivity and precision (F score). As can be seen from Figure 2-8 and Figure 2-9, reciprocal assembly support is more difficult to obtain both with shorter reads, and in the presence of repetitive sequence.

Figure 2-8 F-scores for GRIDSS calls with reciprocal assembly support
At 30x allelic coverage, GRIDSS can detect rearrangement events under most conditions. Performance varies with fragment size and read length with best performance on 75-100bp with long fragment sizes. Relative to the other aligners, bwa mem performs poorly with 36bp reads whilst bowtie2 has worse performance at 75bp or longer. Non-linear scales have been used.

![F-scores for all GRIDSS calls](image)

**Figure 2-9 F-scores for all GRIDSS calls**

At 30x allelic coverage, GRIDSS can detect rearrangement events under all conditions. Performance of at lower coverage levels varies based on the fragment size and read length. GRIDSS is robust to choice of aligner, albeit with a performance penalty for bowtie2 still present at low read depths. Non-linear scales have been used.

GRIDSS was robust to the choice of aligner across the entire input range with only a minor performance penalty for bowtie2 compared to alignment with bwa mem or novoalign. With a 500bp
fragment size, the bowtie2 penalty increased significantly with the cause being traced to a known issue with the default settings of bowtie2 when aligning concordant fragments larger than 500bp\(^2\).

Whilst unimportant for single nucleotide variant (SNV) calling, the library fragment length distribution plays an important role in SV detection with the effect most noticeable at lower levels of coverage. For 2x100bp reads, the sensitivity of GRIDSS at 8x with 500bp fragments significantly exceeds that at 15x with shorter 150bp fragments (90% vs 78%). This trend is present across all read length (36-250bp) and fragment sizes (150-500bp) (Figure 2-9). In general, libraries with longer fragment lengths improve contig length as well as directly increasing the variant spanning read pair coverage hence the signal strength of true variants.

![Figure 2-10](image)

**Figure 2-10 GRIDSS precision**

At low coverage, assembly support at both breakpoint is not present. From 15x allelic coverage onwards, precision of calls supported by only one-sided assembly or with no

\(^2\) [http://sourceforge.net/p/bowtie-bio/bugs/323/](http://sourceforge.net/p/bowtie-bio/bugs/323/)
assembly support drops dramatically, whilst precision of calls with reciprocal assembly support remains close to 100% under all conditions.

2.3.2.1 Contig assembly rate

To determine the coverage required for assembly for a given read length and fragment size, assembly and supporting read counts were extracted for each breakpoint of true positive GRIDSS variant calls. As shown in Figure 2-11, the GRIDSS assembly rate depends primarily on the read length, fragment size, and number of supporting reads. In all cases, increasing the number of supporting reads increases the likelihood of assembly. For shorter read lengths, there is an increased likelihood of both missing assembly as well as fragmented assembly (multiple assemblies for single breakend due to insufficient read overlap). In the extreme case of 2x36bp with 500bp fragments, assembly remains heavily fragmented even at high coverage. Even with such high levels of fragmentation, libraries with longer fragment sizes outperform shorter libraries due to the greater number of supporting reads for a given read depth. It should be noted that even if assembly is fragmented and a single contig cannot be assembled, variant calling is still improved as GRIDSS imposes no penalty for support from multiple assemblies and the assembly contigs can still be more accurately aligned than their shorter constituent reads.
Figure 2-11 Breakdown of assemblies generated by supporting read count

For 100bp onwards, assemblies can be reliably generated from the minimum of 3 supporting reads required for GRIDSS assembly. For shorter read lengths, larger fragment sizes result in fragmented assembly with multiple assemblies mutually supporting the breakpoint. Note that for a given read depth, the expected number of supporting reads increases linearly with fragment size.
2.3.3 Comparison of positional and windowed de Bruijn graph assembly

To compare the effect of choice of assembly algorithm, a windowed assembler using a traditional de Bruijn graph was also implemented in GRIDSS.

Windowed de Bruijn graph assembly is performed using a traditional de Bruijn graph with a dynamically-sized sliding window. The sliding window is sized such that all supporting reads for a correctly assembled breakpoint contigs will be present in the de Bruijn graph when the associated contig is called. Reads supporting putative structural variants are added to the graph in-order in a single pass over the coordinate sorted intermediate files.

An initially empty annotated de Bruijn graph is constructed to which SV-supporting reads are added and removed. Graph edges are determined by kmer adjacency, with each node in the graph containing the following attributes:

- **kmer**: sequence of \( k \) (default \( k=25 \)) bases.
- **weight**: Kmers are weighted by the minimum of the phred-scale base quality score of all bases spanned by the kmer. Total kmer weight is the sum of the weight for each contributing read kmer.
- **reference support**: a kmer supports the reference if all bases contained within that kmer are locally aligned to the reference in at least one split read.
- **expected positions**: for split reads, the expected position is the starting position of the kmer if the split read were fully aligned; for read pairs, the expected position is the position of the kmer if the mate read were to be perfectly concordantly aligned with the read pair from a fragment of median size.

All read bases of split reads and the discordant pair mate reads are split into successive kmers and added to an annotated de Bruijn graph in the order of their alignment position. Since the strand for the read is known from the alignment, only one of the sequence or reverse complement is added.

2.3.3.1 Subgraph tracking

Since only reads supporting putative breakpoints are included in the graph, the de Bruijn graph consists of multiple isolated subgraphs, each supporting a different putative structural variant. Exploiting this property, the minimum and maximum expected position for each connected subgraph
is tracked. When the current genomic traversal position reaches the point where subsequent reads will not support the same breakpoint as reads in a given subgraph (that is, the current position is more than 2 maximum concordant fragment lengths after the subgraph maximum expected position), subgraph assembly is performed, and the entire subgraph removed.

To prevent an unbounded graph size in highly repetitive regions, if the width of a subgraph exceeds a safety threshold (16 times maximum concordant fragment length), subgraph assembly is performed immediately.

2.3.3.2 Subgraph Assembly

The same error correction and graph compression algorithms that are used in the positional de Bruijn graph implementation are used and contigs are called on the error corrected subgraph. The subgraph is divided based on the presence of reference support. For each connected subsubgraph of nodes with no reference support, the longest weighted path is calculated subject to the following constraints:

- The path must start at a successor to a node with reference if possible
- The path must end at a predecessor to a node with reference if possible
- The path must satisfy one of the above criteria if no paths satisfy both

The longest weighted path problem is a well-studied computer science problem which has proven to be NP-hard in the general case, but solvable in linear time for directed acyclic graphs and the standard directed acyclic graph algorithm is used for finding the best path with slight modification. The number of nodes traversed is recorded and if a safety threshold is reached, full path traversal is aborted and path finding falls back to a simple greedy traversal to prevent exponential runtime in degenerate edge cases such as occurs in highly repetitive genomic regions.

Once the best breakend path has been identified, the start and end of the path is extended using longest weighted path traversals restricted to only nodes providing reference support. An assembled contig is created from the underlying kmers. The location and nature of the assembly is determined by the existence of starting/ending reference supporting nodes. How the assembly is treated is determined by whether breakend sequence is flanked on both, one, or neither side by reference supporting sequences:

- both: assembly supports an indel between the expected positions of the breakend flanking kmers.
- one: assembly indicates a large structural variation from the expected position of the flanking kmer to an undetermined location.
• neither: assembly indicates a large structural variation but the exact breakpoint location could not be determined. A breakpoint position confidence interval is calculated from the contributing read pairs.

2.3.3.3 Assembly comparison results

GRIDSS was run using both assembly algorithms on the 50x NA12878 Platinum Genomics data set and assembly rates compared over a range of kmer sizes (10, 13, 16, 19, 22, 25, 28, and 31). Since GRIDSS encodes each sequence kmer within a 64 bit integer using the 2 bit UCSC packedDna encoding scheme (https://genome.ucsc.edu/FAQ/FAQformat.html#format7), the maximum kmer size supported by GRIDSS is a 32mer thus the range of kmers compared is representative of the kmer sizes available to GRIDSS.

Hg19 RepeatMasker annotations were downloaded from UCSC and the assembly rates per top-level repeat class were calculated. Each assembly was assigned to a single repeat class based on the repeat class with the greatest overlap of the assembly position. Assemblies not overlapping any repeat annotated were classified as background. As can be seen in Figure 2-12, the background assembly rate outside of repetitive sequence is lower in positional assembly compared to windowed assembly. Reflecting the robustness with respect kmer size of positional assembly variant calling, the raw assembly rates of positional assembly is robust to choice of kmer size. Windowed assembly lacks this robustness and assembly rates increase at a greater than exponential rate with decreasing kmer size.
Unannotated regions are labelled as background.

Common across both assembly approaches in the increase in the raw assembly rate in regions of repetitive sequence compared to the unannotated background rate. Whilst most repeat classes observe an assembly rate approximate 4-fold higher than the background rate, low complexity, simple repeats, SINE elements and “other” repeats have a 20 fold increased rate of assembly. Due to the repetitive nature of these elements, this increased rate is not unexpected and is only problematic if the resultant assemblies have a high rate of high quality false positives. Figure 2-13 shows that these regions are indeed enriched for high scoring assemblies, but this enrichment is restricted to windowed assembly only.
Over-assembly of low complexity and simple repeats regions results in high confidence false positive calls using windowed assembly, but not positional assembly.

GRIDSS assemblies can be anchored on one side, both sides, or neither side, representing a rearrangement to a remote location, an assembly spanning a small indel, or an assembly with an imprecise breakpoint respectively. Figure 2-14 shows that by breaking down assembly rates according to the assembly anchoring, a high rate of high quality assemblies with flanking anchors is present in windowed assembly, but not in positional assembly.

**Figure 2-13 Assembly quality score distribution by repeat type**
A high rate of flanking anchor assembly using windowed assembly is indicative of intra-window misassembly due to sequence homology.

Figure 2-15 breaks down the flanking anchors assembly further by looking at the assembly type both before and after realignment. For assemblies anchored on only one side, the assembly rate for both positional and windowed assembly are similar, indicating a similar detection capability for large rearrangement events. For intra-window small events, the windowed assembly detection signal is drowned out by the false positive noise caused by sequence homology. For a pre-realignment assembly to have flanking anchors in positional de Bruijn graph assembly, a putative SV read with sequence homology must exist in the expected anchoring position, but with windowed de Bruijn graph assembly, such a read can exist anywhere within the graph assembly window and thus the resultant higher rate of flanking anchors. As a result, the windowed assembly approach as used in GRIDSS is not appropriate for intra-window event calling.
Whilst both windowed and positional assembly has similar detection capability for large events in which only a single side of the event can be assembled, windowed assembly calls orders of magnitude more intra-window events due to misassembly of unrelated reads containing sequence homology. The trend of a reducing post realignment flanking anchor assembly indicates that a kmer size of 40-50 would be required for windowed assembly performance to match that of positional assembly with kmer of 20 for intra-window events.

2.3.4 Performance in repetitive regions

GRIDSS optionally supports the analysis of read alignments with multiple reported mapping positions (described below). However, applications such as precision cancer medicine require high specificity and focus on interpretable or actionable calls, so by default GRIDSS does not consider multi-mapping reads. What separates GRIDSS from callers that do not handle multi-mapping reads is that GRIDSS is able to partially recover multi-mapping read support through assembly without the ten-fold computational and storage overhead of multi-mapping alignment. Since GRIDSS considers multi-mapping reads as unmapped, multi-mapping reads fall into one of two categories: lost reads consisting of reads from read pairs for which both reads are unmapped or multi-mapping, and recoverable reads from read pairs for which one read is uniquely mappable. These recoverable reads are treated as OEAs and, with sufficient coverage, can be assembled into a single contig. Since these contigs are both longer than the read length and have sequencing errors corrected, they are more easily mappable

**Figure 2-15 Assembly rate by anchor type**

[Image of graph showing assembly rate by anchor type]
than individual reads. Many repetitive regions that are not uniquely mappable at the read level are uniquely mappable at the contig level.

Using this process of indirect recovery of multi-mapping reads, GRIDSS is able to maintain a low false discovery rate in repeats other than low complexity, simple, and satellite repeats (See Figure 2-16). RepeatMasker annotations of the lung cancer data indicate that 37% of the high-confidence GRIDSS calls were SVA or LINE translocations.

**Figure 2-16 Variant calls by repeat class**

Breakpoints are broken down by RepeatMasker repeat class annotation. RNA repeat classes have been merged, background indicates no repeat annotation. GRIDSS is able to robustly call structural variants in large repeats but has a higher false discovery rate in low complexity, simple, and satellite repeats.

This approach allows GRIDSS to identify breakpoints even when one breakend consists of entirely multi-mapping reads. On the 778 neochromosome data, GRIDSS was able to correctly identify a breakpoint located in a SINE element that was supported by zero uniquely mappable split reads or read pairs. The 318bp breakpoint contig assembled by GRIDSS was uniquely aligned to the correct SINE...
element (exact alignment match, next closest differed by 6bp) even though every constituent reads was incorrectly aligned.

When compared to multi-mapping aware callers, the GRIDSS approach has both advantages and disadvantages. Since GRIDSS relies on uniquely mapping anchoring reads, GRIDSS is unable to call variants callable by multi-mapping aware callers in which both sides of a breakpoint are multi-mapping at the read level. Conversely, whilst multi-mapping aware callers have a limited number of mapping locations they consider (e.g. VariationHunter defaults to 300 positions), GRIDSS has no such limitation.

2.3.5 Performance on complex somatic rearrangements

To test the performance of GRIDSS on complex genomic rearrangements, we used it to detect rearrangements in published cancer-associated neochromosome datasets [142]. Neochromosomes arise following chromothripsis and multiple circular breakage-fusion-bridge cycles, and contain hundreds of rearrangements. The sequencing data was derived from flow-isolated neochromosomes from well/de-differentiated liposarcoma cell lines: 778 (RRID:CVCL_M808), GOT3 (RRID:CVCL_M819), and T1000 (RRID:CVCL_M809). The reads were primarily 100nt paired end reads, although the 778 cell line data comprised 35, 75 and 100nt long reads. Reads were aligned with bowtie2. GRIDSS was run with default parameters on each of the three samples. Variants outside the published neochromosome contiguous genomic regions (CGRs) were filtered. GRIDSS calls were compared to the published calls, which were predicted using a bespoke discordant read pair method and had undergone extensive manual curation. Calls were considered equivalent if both breakpoints were within 200bp. Events of less than 500bp were filtered as per the methods used in the published call set.

Across the all three samples, GRIDSS showed strong agreement with the published rearrangements with a sensitivity of 98% (1,335/1,361), with only 2.6% supported by only one-sided or no breakpoint assemblies (31, 4 rearrangements respectively). By refining the call set to base pair accuracy, GRIDSS was able to identify breakpoint microhomology in 64% of rearrangements, untemplated inserted sequence in a further 10%, with the remaining 26% of breakpoints characterised by blunt end-joining.

Upon investigation of a number of breakpoints in which GRIDSS and the published call set disagreed concerning the rearrangement partner, a number of compound rearrangement events were identified. GRIDSS was able to identify compound rearrangements involving an additional short fragment from a third genomic location omitted in 5% (64) of the published rearrangements as well as identifying a further 66 compound rearrangements in which the spanning fragment could not be unambiguously placed. The compound rearrangements were identified by GRIDSS assembly contigs.
spanning not only the rearrangement event for which the assembly was generated, but also an additional nearby rearrangement. Since the published call set is based on supporting read pair, these compound rearrangements containing short fragment were misclassified as simple rearrangements, as a sufficient number of read pairs spanned the intervening short fragment.

GRIDSS identified 61 compound rearrangements misclassified by the published call set. 78% of the additional calls made by GRIDSS fell below the 7 read pair support threshold used by the published call set or lay within 1000bp of another rearrangement.

The value of performing variant calling after assembly is highlighted by the rearrangements identified by GRIDSS but not detected by the original discordant read method in the published rearrangement set. The increased sensitivity of GRIDSS allowed 551 additional rearrangements to be identified of which 364 either fell below the 7 read pair threshold used in the construction of the published call set.
or lay within 1000bp of another rearrangement. Of these, 233 rearrangements were supported by reciprocal breakpoint assemblies and a further 112 by single-sided assembly.

2.3.5.1 Comparison to other callers on neochromosome call set

To compare GRIDSS to the other callers on the neochromosome call set, all callers were run on a subset of the input. As not all callers are capable of handling multiple libraries containing different read lengths, callers were compared using only the 2x100bp libraries. Callers using aligned reads as input were run against both bwa mem and bowtie2 aligned reads with the result with the highest sensitivity used. Calls were matched if both breakends were within 180bp, the event sizes differed by no more than 25%. Events of less than 500bp were filtered as per the methods used in the published call set. Unmatched calls outside of the published Contiguous Genomic Regions (CGRs) were ignored. Results were aggregated across the 3 data sets. GRIDSS identified the most calls in the published call set, with up to 74% overlap between other callers, and the GRIDSS High Confidence call set (Table 3).

<table>
<thead>
<tr>
<th>Caller</th>
<th>Published call set concordance</th>
<th>Overlap with GRIDSS High Confidence</th>
<th>Additional Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRIDSS</td>
<td>93%</td>
<td>100%</td>
<td>20787</td>
</tr>
<tr>
<td>BreakDancer</td>
<td>17%</td>
<td>25%</td>
<td>4454</td>
</tr>
<tr>
<td>Cortex</td>
<td>1%</td>
<td>1%</td>
<td>10</td>
</tr>
<tr>
<td>DELLY</td>
<td>91%</td>
<td>58%</td>
<td>8019</td>
</tr>
<tr>
<td>HYDRA</td>
<td>27%</td>
<td>28%</td>
<td>4742</td>
</tr>
<tr>
<td>LUMPY</td>
<td>91%</td>
<td>71%</td>
<td>2094</td>
</tr>
<tr>
<td>Pindel</td>
<td>40%</td>
<td>51%</td>
<td>916</td>
</tr>
<tr>
<td>SOCRATES</td>
<td>86%</td>
<td>74%</td>
<td>45939</td>
</tr>
</tbody>
</table>

Table 3 Comparison of structural variant callers to the published, and the GRIDSS high confidence call sets

Comparison based on variant calling on 2x100bp libraries only. Additional calls are those not in either the GRIDSS high confidence call set or the published call set.
2.3.6 Performance on multi-mapping read alignments

To assess the performance of GRIDSS using multiple mapping locations, the NA12878 analysis was repeated with the aligner configured to report multiple mapping locations. Paired FASTQ files were extracted from the NA12878 Illumina Platinum Genomics BAM file and realigned using bowtie2 with up to 100 alignment positions reported for each read (using parameters “-k 100 --fast-local --local -X 1000”). Both coordinate and read name sorted BAM files were produced. GASVPro, HYDRA-Multi, and GRIDSS were run on the multi-mapping input file. In the case of GASVPro, the read name sorted BAM file was annotated with the IH SAM tag using GRIDSS. ComputeSamTags and the BAM file separated into two files (one containing uniquely mapped reads, and another containing multi-mapping reads), as per the GASVPro documentation. As bowtie2 does not report a meaningful MAPQ score when reporting multiple alignments, GRIDSS variant calls were scored based on read counts with a minimum of 2 reads required to support a variant, and 13 read required to be considered high confidence. To ensure multi-mapping reads were considered by GRIDSS, the minimum required MAPQ was reduced to 0, the maximum coverage increased to 50,000, and the minimum read entropy reduced to 0. To ensure variant calls in repetitive sequence were called, the long read mapping criterion requiring a mapping quality score of greater than zero was dropped.

As can be seen in Figure 2-18, the sensitivity of GRIDSS was improved by 26%, but with reduced specificity. Results for GASVPro could not be obtained as after 1 month execution had not completed. Results for HYDRA-Multi were unexpectedly poor with a multi-mapping sensitivity less than 10% of single-mapped sensitivity and a false discovery rate of over 99.9%. The issue was reported to the developer, but not resolved and HYDRA-Multi results were excluded.
Multi-mapping alignment was performed using up to 100 alignments per read. GRIDSS variant calls were compared to both the Mills et al. validation call set (a) and PacBio/Moleculo long read orthogonal validation data (b).

Although we were unable to obtain meaningful results from a multi-mapping aware caller to provide a direct comparison with other algorithms, a comparison can be made to independently reported results. Using both multi-mapping reads and trio information, Hormozdiari et al. reported that of the 5,000 best-supported predictions made by VariationHunter-CommonLaw on NA12878, the Mills et al. validated deletion calls supported 1723. Using only the best read alignment, GRIDSS significantly outperformed VariationHunter with 2086 of the 2953 deletions in the GRIDSS high confidence call set supported by Mills et al., increasing to 2,240 in the top 5,000 of all GRIDSS calls. When using input from a multi-mapping aligner, GRIDSS identified with 2017 of the 3036 high confidence deletions supported by Mills et al. reducing to 1,102 in the top 5,000 calls. This loss of specificity is mitigated by a 26% increase in sensitivity with 5139 and 6506 deletions supported by long reads in the single alignment and multi-mapping alignment call results respectively.

A number of phenomena contribute to these results. Firstly, even without multi-mapping, GRIDSS is able to identify many variants in repetitive regions by virtue of the genome-wide break-end assembly performed prior to variant calling. As demonstrated in the context of tumour neochromosomes (see 2.3.5 Performance on complex somatic rearrangements), GRIDSS was able to identify a breakpoint into a SINE element in which every supporting read originating from the SINE element was incorrectly mapped, due to the novel assembly approach taken by GRIDSS. With over 80% of the GRIDSS calls
identified as true positives in regions annotated as repeats by RepeatMasker, GRIDSS is able to recover a large number of repeat variants without requiring multi-mapping alignment. Secondly, the incorporation of split reads allows GRIDSS to detect smaller events than can be detected by multi-mapping read pair analysis alone. To our knowledge, GRIDSS is the first structural variant caller to incorporate split read alignments into a multi-mapping aware caller. Thirdly, unlike aligners such as mrFAST, bowtie2 multi-mapping reports the first 100 potential alignment locations but does not subset these on edit distance. As a result, bowtie2 will report alignments with considerably different edit distances (in some cases 3 to 10) thus including alignment locations that are highly unlikely to be correct. As GRIDSS has been designed to be aligner agnostic and only require alignments to conform to the SAM specifications, this results in an increased false discovery rate for bowtie2. Thirdly, the realignment of assembly break-end contigs is performed using bwa mem, which reports only the best mapping for the contig. In the case of ambiguous contig alignments, this results in assembly support for only one of the potential locations. Finally, the reporting of variants in large, truly repetitive sequence is highly problematic as unambiguous placement of a variant is in many cases not possible since the signal in sequencing data is limited by the fragment size of the input library. When using an output format based on genomic coordinates such as VCF, a variant caller able to correctly detect the presence of a variant in a large perfect repeat present in multiple locations is faced with a choice: report the variant in a single location (thus likely to be a false positive); or report all positions (thus reducing specificity). This confounds both the downstream analysis and the validation of such calls. Further work is needed on downstream analysis and in standardising the reporting of variants that cannot be unambiguously placed.

2.3.7 Validation on lung cancer

Lung tumours were collected from two stage II lung cancer patients (LP1-43 and LP2-33) who underwent surgical resection at the Austin Hospital, Melbourne, Australia. Tumour biopsies were taken for snap freezing in liquid nitrogen and for fixation in 10% buffered formalin. A pre-operative and a series of post-operative blood samples were collected from the participating patients. Plasma was separated from whole blood within 3 hours since its collection using two centrifugation steps, 800 and 1600 rcf for 10 minutes each. Buffy coat was also collected to use as a germline DNA control. All collected plasmas and buffy coats were then stored at -80°C until DNA extraction. Informed consent was obtained from the two patients and the study was approved by the Institutional Review Board of Human Research Ethics Committee.

Primary lung cancer tumours were sequenced to 40x coverage on an Illumina HiSeq generating paired 100bp reads. Reads were aligned using bowtie2 version 2.2.5 with --qc-filter -X 1000 --local command
line parameters and duplicates marked using Picard tools. Variants were sorted by quality score reported by GRIDSS and the following filters applied:

- Variants not in the GRIDSS high confidence call set were filtered.
- dGV (Database of Genomic Variants) hits within 10bp were filtered.
- Alternate contig variants were filtered.
- Variants less than 1Mb were filtered.
- Translocation under 5kbp were filtered.
- Variants without split read, read pair, and assembly support were filtered.
- Telomeric and centromeric variants were filtered.
- Variants with a NCBI blast hit (all human sequences) spanning the breakpoint were filtered.

Candidate rearrangements were manually inspected using IGV and the 15 highest scoring variants with a change in copy number or SNV allele frequency supporting somatic origin were selected as candidates for validation. Of these 15 candidates, 8 were selected for validation (4 from each patient) based on the ease of PCR primer design.

Genomic DNA was extracted from fresh frozen and formalin-fixed tumours, and buffy coat using the DNeasy Blood and Tissue Kit (Qiagen, Hilden, Germany). Plasma DNA was extracted from 2 mL of plasma using the QIAamp Circulating Nucleic Acid kit (Qiagen) by eluting twice with 50 µL of ACE buffer.

Primers were designed to flank the rearrangement breakpoints and to generate less than 100 bp amplicons for fragmented formalin-fixed and plasma DNA. The validation experiments were carried out on a CFX Connect (BioRad). PCR reaction mix in a final volume of 20 µL contained; 1× PCR buffer, 2.5 mM MgCl₂, 100 nM each primer, 200 µM of dNTPs, 5 µM of SYTO 9, 0.5 U of HotStar Taq DNA polymerase (Qiagen) and 10 ng of genomic DNA. The cycling and melting conditions were as follows; one cycle of 95°C for 15 min; 55 cycles of 95°C for 10 s, 68-60°C (gradient PCR) for 20 s and 72°C for 30 s, followed by one cycle of 97°C for 1 min and a melt from 70°C to 95°C temperature rising 0.2°C per second.

The presence of tumour DNA containing the validated genomic rearrangements was tested in a series of plasma DNA samples using the QX200 droplet digital PCR system (BioRad). PCR mix was prepared with EvaGreen ddPCR supermix, 100 nM each primer and 2 µL of plasma DNA in a final volume of 23
µL. The manufacturer’s instruction was used for droplet generation, PCR cycling and droplet reading. Analysis of the ddPCR data was performed with QuantaSoft analysis software (Bio-Rad).

2.3.8 HCC1395 cell line genomic breakpoints identification

HCC1395 tumour DNA-Seq sequences were downloaded for SRP022140 from sra.dnanexus.com. Of the nominal 63x coverage, effective coverage was only 21x as SRR892402, SRR892256, and SRR866931 contain identical sequences, as do SRR866925, SRR892454, and SRR892651.

Reads were aligned using bwa mem with default parameters. GRIDSS 0.10.4 was run with default parameters. GRIDSS calls were matched to the 240 junctions (including the 102 missing validation probes), and to all 26 genomic breakpoints with 250kbp and 1kbp margin respectively using https://github.com/PapenfussLab/sv_benchmark/blob/master/R/HCC1395.R. Matching calls were then manually filtered to exclude variants with position or orientation not consistent with the gene fusion (see Appendix A). Putatively causative variants were manually filtered by inspection in IGV. Of the 3 genomic breakpoints not identified by GRIDSS, zero split read or read pairs supporting SHISA5/RP11-24C3.2 could be found, and NOM1/MNX1-AS2 and CCDC117/CTA-292E10.6 were each supported by only a single read pair able to be explained as the sequencing of a long fragment. In addition, GRIDSS was able to find 3 additional causative genomic breakpoints not previously identified. Concerningly, neither NOM1/MNX1-AS2, CCDC117/CTA-292E10 nor MAVS/PANK2, WARS/DEGS2, and RP11-85K15.2/BAZ1A are able to explain the either corresponding fusion transcript. In all these cases, read-through gene fusion not only is still required to explain the transcript, but the transcript can be explained purely though read-through gene fusion without the presence of any genomic breakpoint.

2.3.9 Variant scoring model comparison

In addition to the empirical reference likelihood model used as the GRIDSS default, we investigated three additional models (see Figure 2-19). All models result in similar results with the empirical reference likelihood models performing 4% better that read counts when performance is defined as the area under the curve truncated at 1000 false positives.
Figure 2-19 ROC curve for alternative GRIDSS models

ROC curves generated from Illumina Platinum Genomics 2x100bp 50x NA12878 data using Moleculo/PacBio long read truth set.

2.3.9.1 Read Count
This model weights all reads equally. As each read is given a score of 1, this matches the read counting used by other calls for ranking variants.

2.3.9.2 MAPQ
This model ignores the empirical library distributions and weights according to the Phred-scaled likelihood of the read being correctly mapped.

2.3.9.3 Empirical log-likelihood ratio
Under this model, we compare the likelihood of a structural variant $V$ between genomic locations $a$ and $b$ of variant allele frequency $\alpha$ that best matches the data, and the null reference allele hypothesis $R$. Under the null (reference allele) hypothesis, no structural variations exist and all mapping errors are assumed to be uniformly distributed noise. Variants are scored according to the log likelihood ratio.
(LLR) of the variant and the null reference models. In general, the probably of evidence e given allele A is of the form

$$\Pr(e|A) = \Pr(M|A) \cdot \Pr(e|M \land A) + \Pr(\overline{M}|A) \cdot \Pr(e|\overline{M} \land A)$$

Where M is a correct mapping of the evidence, and \(\Pr(e|M \land A)\) and \(\Pr(e|\overline{M} \land A)\) are determined by the empirical distribution of the evidence. When the mapping is incorrect, the read is uninformative with \(\Pr(e|\overline{M} \land A)\).

The empirical distributions of read and read pair mappings for each library are calculated so that probability distributions \(P_{sc}(l)\) of a mapped read having a soft clip of length \(l\), \(P_{rp}(f)\) of a read pair mapping with inferred fragment length \(f\), and the probabilities \(p_d\) of read pair mapping discordant, and \(p_u\) of a single read of the pair unmapped are known.

For split reads \(sr\), \(\Pr(M|A) = q_a q_b\) and \(\Pr(e|M \land A) = \Pr(e|\overline{M} \land R) = P_{sc}(l_a)\), where \(l\) is the length of the of soft clip and \(q_l = 1 - 10^{-\text{mapq}(e)/10}\) is the correct read mapping probability reported by the aligner at \(i\).

$$\log \left( \frac{\Pr(sr|V)}{\Pr(sr|R)} \right) = \log \left( \frac{\Pr(M|V) \cdot \Pr(sr|M \land V) + \Pr(\overline{M}|V) \cdot \Pr(sr|\overline{M} \land V)}{\Pr(M|R) \cdot \Pr(sr|M \land R) + \Pr(\overline{M}|R) \cdot \Pr(sr|\overline{M} \land R)} \right)$$

$$= \log \left( \frac{q_a q_b \cdot P_{sc}(0) + (1 - q_a q_b) \cdot P_{sc}(l_a)}{q_a q_b \cdot P_{sc}(l_a) + (1 - q_a q_b) \cdot P_{sc}(l_a)} \right)$$

$$= \log \left( \frac{q_a q_b \cdot P_{sc}(0)}{P_{sc}(l_a)} + (1 - q_a q_b) \right)$$

Similarly, discordant read pairs \(dp\) are modelled as \((q_a, q_b, f)\) with \(\Pr(M|A) = q_a q_b\) and \(\Pr(e|M \land A) = p_d + P_{rp}(f_A)\).

$$\log \left( \frac{\Pr(dp|V)}{\Pr(dp|R)} \right) = \log \left( \frac{q_a q_b \cdot p_d + P_{dp}(f_V)}{p_d + P_{dp}(f_R)} + (1 - q_a q_b) \right)$$

2.3.10 Performance Optimisation

2.3.10.1 Record caching

To assemble a contig supporting a putative structural variant at a given genomic location, the full read sequence of all supporting reads is required. By sorting the input and intermediate alignment files according to alignment position, retrieving one half of a DP or SR can be done efficiently as they are stored contiguously within the file. To improve the retrieval speed of the other half of a DP or SR, an additional copy is made and sorted by the genomic coordinate of the partner. Simultaneous traversal of both files enables both halves of a DP or SR to be accessed at both mapping locations with purely
sequential file access. Actual disk storage required for all intermediate file varies considerably between data sets but is typically 5-25% of the size of input files.

2.3.10.2 Record streaming

For compatibility with existing tools and file formats, and to allow visualisation of intermediate results, reads are stored using the BAM file format. The SAM/BAM format defines the coordinate sort order position of a mapped read as the position of the first aligned base of the read. The order in which evidence is processed in GRIDSS is by the starting position of the 95% confidence interval of putative breakpoint positions. For split reads this is the position of the mapped base adjacent to the split, which only matches the coordinate sort order if the soft clip occurs at the start of the read. Similarly, depending on the strand to which the local read is aligned, the start of the breakend interval for read pairs occurs either maximum expected fragment size – mapped read length bases before the first mapped base, or after the final mapped base.

The distance between the breakend start position and the position of the first base is bounded. For soft clips, it is bounded by the maximum mapped read length, and for read pairs by the maximum expected fragment size and the maximum mapped read length. This allows evidence to be reordered in-memory by buffering all reads within max(maximum expected fragment size, maximum mapped read length) of the current position. GRIDSS implements this buffer as a priority queue sorted by breakend start position.

2.3.11 Runtime Performance Breakdown

By breaking down runtime performance of GRIDSS by processing stage, bottlenecks and multithreading effectiveness can be ascertained. Both external alignment and assembly scale with multiple threads and, in a multithreaded environment, execution time is dominated by single-threaded processes that traverse the raw input file (Figure 2-20). Since assembly multithreading is performed at the reference chromosome level, the maximum theoretical improvement is limited to the maximum chromosomal assembly runtime, for NA12878 50x WGS, the limiting chromosomes are 1 and 2 taking 12 and 15 minutes respectively, with all other chromosomes completing in under 9 minutes.
Figure 2-20 Breakdown of GRIDSS multi-threaded runtime

Runtime based on Illumina Platinum Genomics 2x100bp 50x NA12878 data with ENCODE DAC blacklist. Multithreaded execution time is dominated by the three traversals of the input file.
Figure 2-21 Positional error in variant position relative the simulated event position

Positional error in variant position relative the simulated event position. A green histogram allowing for caller-reported micro-homologies has been overlayed over the orange histogram considering only the called position. GRIDSS, cortex, and manta are able to correctly report micro-homologies. Except for the high positional error of lumpy, the separation of callers into discordant pair and split read/assembly based callers can be inferred from the error profile.
2.3.11.1 Blacklist
Since reads with low mapping quality are filtered before assembly, multi-mapping reads in repetitive genomic regions do not affect assembly performance. Other sequences such as satellite, centromeric and telomeric repeats have a considerable effect on assembly performance. These are regions that may be poorly assembled in the reference or unrepresentative of individuals. Alignments to these regions are likely dominated by reads from paralogous sequences not present in the reference. Without any explicitly blacklisted regions, single-threaded assembly of NA12878 50x WGS takes 504 minutes which is reduced to 145 minutes if regions in the ENCODE DAC blacklist are ignored.

2.3.11.2 Assembly Memory Consumption Breakdown
Since the assembly window is extended to ensure the highest scoring contig can have no supporting reads overlapping with any partially constructed contig, the assembly memory construction is theoretically bounded by only the size of the input. This theoretical upper bound can be realised through the creation of an input in which each putative assembly contig is followed by a nearby putative assembly contig with greater support. In practice, memory consumption is much lower, as outlined in Figure 2-2221. The assembly graph contained an average of only 62.7 reads when calling a putative forward contigs on chr12 of the NA12878 platinum genomics 50x 2x100bp data set. The majority of these contigs were supported by singletons with only 42,855 of the 730,000 putative contigs called passing basic filtering (at least 3 reads support, not aligning to reference). With an average of 3770 nodes, GRIDSS assembly memory consumption is dominated by the trackerLookup hash table which records which positional de Bruijn nodes are supported by which reads.

Whilst most data structures contain relatively few records, peak size for all structures is at least two orders of magnitude larger than the mean. These peaks generally occur in repetitive regions of the genome, with the largest peaks in centromeric regions and nuclear mitochondrial DNA (NUMTs) with corresponding increases in runtime. In extreme cases (such as >100,000x coverage of some centromeric sequences in 50x WGS data sets), a high-coverage 1kbp window will take longer to assemble than an entire chromosome. Such bottlenecks are mitigated through the blacklisting of the extreme high-coverage regions, and by the downsampling performed at discordant coverage extrema.
Distribution of number of entries in GRIDSS streaming assembly data structures per assembly call. Counts are taken from Platinum genomics 2x100bp 50x NA12878 chr12 forward assembly. Memory consumption is dominated by the hashtable linking graph nodes.
with their supporting reads as entries not only track every positional kmer for the read, but must remain in this table from first processing of the read until a contig containing the read is called.

2.3.11.3 Assembly graph size constraints

Although the locally maximal contig for any given read is the globally maximal contig, the number of nodes loaded before a maximal path is reads is unbounded. For any given read, a nearby successor read could form part of a higher scoring contig, which in turn could have a higher scoring nearby successor. As coverage increases, the likelihood of these successors existing also increases. When tested on 500x coverage WGS data, these chains were up to 10Mb in length.

Whist the priority queue of loaded contigs sorted by score enables maximal contigs to be identified, it does not ensure that the earliest maximal contig is at the head of the queue when the graph is mutated due the contig deletion. Take the scenario of contigs A1, B2, C3 and D2, with the AB, BC and CD pairing close by. Calling A requires B to be loaded, which requires C to be loaded which will load D as part of the graph traversal. If the maximal C is called which results in the removal of C and B from the graph, A1 and D2 will remain. Using a priority queue, A will not be assembled until D is called, even if A and D cannot contain the same reads. For high-coverage data, this results large number of low-scoring contigs remaining in the graph for extended periods of time. To constrain memory consumption, contigs more than 10 times the maximum expected fragment size from the frontier are called whenever the width of the graph exceeds 100 times the maximum expected fragment.

2.4 Conclusion

In this chapter, I have presented GRIDSS: the Genome Rearrangement IDentification Software Suite. This structural variant caller performs genome-wide break-end assembly of soft-clipped reads, split reads, discordant read pairs, and one-end anchored reads pairs, followed by variant calling based on assembly contig, split read, and read pair evidence combined. I have shown that this approach significantly outperforms existing variant calling approaches on both simulated and human cell line data due to the novel break-end assembly approach taken by GRIDSS.

I have shown that a positional de Bruijn graph assembly approach in which the read alignments constraints are encoded in the assembly graph itself, is more robust to repetitive sequences that the equivalent windowed de Bruijn graph assembly in which reads alignment information is only used for the selection of reads to assemble. I have demonstrated that through the use of data streaming and graph compression, genome-wide break-end assembly using a positional de Bruijn graph is computational feasible on commodity hardware.
Chapter 3 Comprehensive evaluation and characterisation of structural variant calling software

In this chapter, I undertake a comprehensive benchmark of general purpose SV callers across a wide range of read lengths, read depths, library fragment sizes and aligner choices to carefully evaluate the strengths and weaknesses of SV calling methods and software. I show that different implementations of the same algorithm differ in their usability, sensitivity, and specificity; that reference genome-based simulation studies vastly over-estimate SV calling performance; that blacklisting/variant filtering is critical for real data; and that better sequencing (longer read length/higher coverage) can, for some callers, result in dramatically worse SV calling performance. I also provide recommendations on which tools to use in different sequencing parameter regimes, and recommendations to SV method developers. An accompanying interactive website for interactive inspection of the results is available online at http://shiny.wehi.edu.au/cameron.d/sv_benchmark/.

3.1 Introduction

Structural variants (SVs) are polymorphisms or genomic rearrangements where the organisation of the genome differs from the typical structure or reference genome. In the case of genomic rearrangements, they involve double stranded DNA breakage and repair. Typically, the term SV is used for events that are greater than 50bp in size[143]. These can include large insertions (including transposons), inversions, balanced or unbalanced translocations, and amplifications and large deletions. Understanding structural variation is important in the study of population diversity, cancer [e.g. 131, 141, 142] and other diseases [144, 145].

Since 2010, over 40 short read sequence-based general purpose structural variant callers have been published (Table 1). Although a number of reviews have been published describing the theoretical algorithmic advantages and weaknesses of structural variant calling approaches [82, 112-119], comparative evaluation and benchmarking of structural variant calling software has been limited. Apart from the comparisons published with each caller invariably reporting results favourable to the software being published, benchmarking studies have been limited to SNV/small indel callers [120-125], or CNVs [20].

To assist in the development of new methods, and to determine the best existing methods to use on large cohort studies such as the 1000 Genomes project [141, 146], a number of consortia are developing high-quality reference call sets. The Human Genome Structural Variation Consortium is working on a set of comprehensive structural variant call sets which include variants in difficult to call regions of the genome[147]. ICGC-TCGA have taken a different approach to somatic variant calling
and, to determine the best variant calling software to use, have crowdsourced putative variant call sets through submissions to the ICGC-TCGA DREAM Mutation Calling challenge:

The ICGC-TCGA DREAM Genomic Mutation Calling Challenge is an international effort to improve standard methods for identifying cancer-associated mutations and rearrangements in whole-genome sequencing (WGS) data. The goal of this somatic mutation calling Challenge is to identify the most accurate mutation detection algorithms, and establish the state-of-the-art. The algorithms in this Challenge must use as input WGS data from tumour and normal samples and output mutation calls associated with cancer.[148]

In this challenge, teams are invited to submit somatic mutation calls from analysing both in-silico simulated tumours and tumour cell lines with separate sub-challenges for SNV [149], indels and SVs calling. This will result not only in a benchmark of somatic callers, but will provide an extensive validated truth set of somatic variant calls. Unfortunately, until the completion of such efforts, the lack of unbiased comprehensive structural variant truth sets is an impediment to evaluation of structural variant callers on cell line data.

Also lacking from the existing literature is a comprehensive evaluation of structural variant calling software across a wide range of data sets. Publications of new structural variant calling software typically include a comparison to 2-5 existing callers. Unfortunately, since variant calling performance can vary with both sequencing choices such as read length, read depth, library fragment size as well as bioinformatics choices such as the processing pipeline and alignment software, it is relatively easy to present results biased, even unintentionally, through the use of a data set on which one or more callers perform particularly poorly. Compounding this is the high variance in the reliability and quality of the implementation of the underlying algorithms in the published software. Whilst software such as GATK [56] is supported by a team of full-time professional software developers, most bioinformatics software is not written by professional software engineers. As a result, structural variant calling software varies wildly in terms of usability, reliability, and correctness of implementation.

To evaluate these shortcomings, this chapter presents a comprehensive benchmark of general purpose structural variant calling software across a wide range of read lengths, read depths, library fragment sizes and aligner choices based on usages typical of an independent user not involved in the development of the software.

I have intentionally designed this study to evaluate structural variant callers as they would be used by a researcher expanding the range of variants considered in their study from SNV/small indel to include structural variants. It is assumed that the researcher has basic familiarity with linux command-line
usage and is able to execute a basic bioinformatics pipeline according to provided user documentation. No familiarity with either the underlying structural variant detection methods, or the usage or implementation details of any structural variant callers is assumed. This scenario is a critically important in real-world usage of publicly released bioinformatics software and reflects the actual gain from the public release of the software. As such, this evaluation and characterisation of structural variant calling software reflects the likely conditions under which the software is expected to be used, not the idealised best-case results able to be obtained by the authors of any given tool.

3.2 Methods

To evaluate methods for SV calling, a comprehensive in-silico simulation across a wide range of sequencing parameter choices and evaluation on a set of well-characterised cell lines has been used.

3.2.1 In-silico data sets

To determine the limitations of SV calling algorithms and software implementations, comprehensive in-silico evaluation of callers was performed. 5 variant types were simulated: insertions, deletions, inversions, tandem duplications, and breakpoints/translocations. To allow precision and recall to be calculated per variant type, each data set contained only heterozygous events of a single type. For insertion, deletion, inversion and tandem duplication events, an equal number of events of sizes 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 16, 20, 24, 28, 32, 48, 64, 80, 96, 112, 128, 160, 192, 224, 256, 288, 320, 512, 1024, 2048, 4096, 8192, 16384, 32768, and 65536bp were inserted for a total of 18,000 events of each type. Events were placed in order at the first available location with at least 2,500bp separation from any other event or ambiguous reference base. Translocations were simulated by fragmenting chr12 into 2,500bp fragments and randomly reassembling 10,000 of the resultant fragments.

Since not all variant callers are capable of the detection of inter-chromosomal events, all events were simulated on a single chromosome. Human hg19 chromosome 12 was chosen as it has close to the median chromosome size and GC content of the human genome, is rich in oncogenes, has been previously used for similar simulations, and all auxiliary reference files required by the variant callers were available for hg19.

Art 1.51 [150] using a MiSeq error profile was used to generate simulated paired-end reads of lengths 36, 50, 75, 100, 150, and 250bp at read depths of 4, 8, 15, 30, 60, and 100x mean coverage, from fragment sizes of 150, 200, 250, 300, 400, and 500 ±10% base pairs. Whilst reads were simulated exclusively from hg19 chr12, reads were aligned against the full hg19 reference genome.
When a result for a variant caller is presented without specifying an aligner, the result corresponds to the result for the most sensitive aligner for the given caller, event type, read depth, read length and fragment size.

3.2.2 SV Callers

The choice of SV callers chosen is critical to the evaluation process. To ensure tools were evaluated based on their intended usage, only general-purpose DNA-Seq SV callers were considered with specialised callers such as transposable element, viral insertion, SNV/small indel, or RNA-Seq gene fusion callers excluded from consideration. To ensure that popular tools were included, and the callers selected were a representative cross-section of the algorithmic approaches used for SV detection, multiple criteria were used for the selection of callers. Firstly, Web of Science citation counts were used as a proxy for tool popularity. Citation counts were retrieved for each tool and variant callers were included if amongst the 20% most highly cited, or amongst the 20% with the highest yearly citation rate. Secondly, the algorithmic approach of each caller was determined (see Table 1) and the tool with the highest yearly citation rate for each approach was added to the list. For tools with multiple publications such as GASV/GASVPro, VariationHunter/VariationHunter-CR/VariationHunter-CommonLAW and HYDRA/HYDRA-Multi, only the most recently published version was selected.

Using these criteria, the following SV callers were selected: VariationHunter-CommonLAW 0.0.4, GASV-Pro 20140228, Pindel 0.2.5b6, BreakDancer 1.3.5, HYDRA-Multi 0.5.2, CREST 0.0.1, DELLY 0.6.8, cortex 1.0.5.14, SOCRATES 1.13, LUMPY 0.2.11, CLEVER 2.0rc3, GRIDSS 0.11.5, SOAPsv, and manta 0.29.6. Although highly cited, BreakPointer was excluded as dRanger, the upstream calling software required for usage of BreakPointer in the Broad pipeline, was not publicly available.

In addition to the structural variant callers, a samtools/bcftools pipeline was included as representative of a typical SNV/small indel calling pipeline.

3.2.3 Variant Calling Pipeline

For each data set, the paired FASTQ input files were aligned to the human reference genome (hg19) using bwa mem, bowtie2, and mrFast. BAMs sorted by chromosome and read name were generated for each output BAM.

For each variant caller, a shell script was created to generate a variant calling shell script for each data set. Using the input format required by the variant caller (paired FASTQ, chromosome sorted BAM, read name sorted BAM, or mrFAST DIVET), variants were called based on the recommended settings. Recommended settings were taken from the usage message received when executing the program with invalid arguments, the user guide, README, software website, or publication in that order of
preference. For variant callers that do not output VCF files, a python conversion script was created to convert the output format to VCF 4.2 notation. Runtime performance was measured using the unix `time` command on the variant calling shell script. Scripts were executed on a dual socket Xeon E5-2690v3 with 512GB of memory.

In line with the expected usage, software authors were not contacted directly regarding recommended settings. If a fatal error was encountered prohibiting the successful execution of the software, a message was posted to a publicly usable mailing list and an issue was raised on a publicly usable issue registry. Read-only issue registries (such as those hosted on google code and not migrated to github) were not considered publicly usable. For each program, a professional software engineer was allocated 2 days to generate the variant calling script and create any reference files required by that program. If after two days a working script could not be created, the program was deemed to have insufficient usability for widespread usage and was excluded.

When determining whether a variant call matches the expected call, matching was based on the set of breakpoint calls for the underlying variant. VCF variant calls were converted to sets of breakpoint calls using the R StructuralVariantAnnotation package (http://github.com/PapenfussLab/StructuralVariantAnnotation). Breakpoints were considered matching if both the start and the end breakpoint position were within 200bp of the true positions, and the sizes differed by at most 25%. For inversion events, both of the underlying breakpoints required a match for the inversion to be considered matching. When the called variant position was ambiguous due to microhomology or inexact due to read-pair only variant support, the breakpoint was considered matched if a match was found at any position in the interval of ambiguity. As BreakDancer does not report the breakpoint orientation of variant calls, breakpoints were considered to match even if their orientations did not. Variant calls were considered high confidence if no VCF filters are applicable to that variant and were considered low confidence if the variant has failed any of the VCF filters applied by the variant caller itself.

All scripts used in this paper are available at http://github.com/PapenfussLab/sv_benchmark

3.2.4 Cell line evaluation

To evaluate how the theoretical performance of SV calling algorithms and software implementations translates to actual data, variant callers were evaluated on well-studied human cell lines. As existing SV truth sets [140, 141] would bias results towards the callers used to generate the call sets, variant callers have been evaluated based on their consistency with data from long read technologies. Variant calls were considered consistent with the long read data if the variant was supported by 3 long reads.
Spanning long read support required the long read to contain an indel with size ±25% of the called size with positional error less than half the indel size (up to a maximum error of 200bp). Similarly, split long read support required both split read alignment locations to be within 200bp of the variant call locations, and the event sizes to be within ±25%.

For the Coriell Cell Repository NA12878 reference cell line, 50x coverage PCR-free 2x101bp reads from a HiSeq2000 were obtained from Illumina Platinum Genomes projects (https://basespace.illumina.com/s/Tty7T2ppH3Tf), 4 replicates of 35x coverage 2x151bp reads from an Illumina HiSeq X Ten sequencing system were obtained from Kinghorn Centre for Clinical Genomics Sample Sequencing Data (http://www.garvan.org.au/research/kinghorn-centre-for-clinical-genomics/clinical-genomics/sequencing-services/sample-data), and PacBio and moleculo long reads were obtained from the 1000 Genomes Phase 3 SV group (http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/phase3/integrated_sv_map/supporting/NA12878).

3.2.5 Interactive visualisation

To aid in the visualisation and interrogation of structural variant caller performance, an interactive graphical website has been developed. This tool was developed using the R Shiny framework. To remove the necessity of loading the entire 130GB data set into memory, all intermediate and final results were cached using the R.cache package. Cached values for each combination of UI options affecting the call or truth sets were pre-computed offline before being transferred to the shiny server. The interactive website for visualisation of the benchmarking results is available at http://shiny.wehi.edu.au/cameron.d/sv_benchmark.

3.3 Results

3.3.1 Method selection and data simulation

We first selected 10 SV callers though a rigorous process (see Methods for details) for testing. To assess the performance of these tools under a wide range of sequencing parameters, we developed a comprehensive, multi-dimensional simulation. For each type of SV (insertions, deletions, inversions, tandem duplication, intra-chromosomal translocation), we simulated from 10,000 (intra-chromosomal translocations) to 18,000 (all other SV types) heterozygous events with sizes from 1bp to 64,000bp on chr12 (hg19). We repeated this for a variety of simulated read lengths (35, 50, 75, 100, 150, 250), fragment sizes (150, 200, 250, 300, 400, 500), and coverage levels (4x, 8x, 15x, 30x, 60, and 100x), and analysed this data using multiple aligners and SV calling tools.

For typical resequencing parameters (2x100bp, 300bp mean fragment size), some trends are immediately apparent (Error! Reference source not found.):
• Overall, the size range of detectable events is in general agreement with theoretical expectations.
• For deletions, callers relying on DP evidence as the primary signal can detect events larger than 100bp, callers using only SR evidence can detect events larger than 30-50bp, and callers performing de novo assembly, OEA split read analysis or incorporating indel-containing reads are able to detect deletions across the full size range.
• As has previously been observed [46], the order in which evidence is applied can be an important factor; DELLY applies DP followed by SR evidence, thus limiting detection capability to that of a discordant read pair only caller.
• For insertions, DP-based callers are able to detect events within a narrow 50bp window around the 150bp event size, soft-clipped split read callers can detect 16-50bp events, OEA split read callers can detect events up to 64bp, break-end/breakpoint assembly can extend the detection out to up to 200bp. Only de novo assembly can detect large insertions of novel sequence.
• Theoretically, the trends for inversions and tandem duplications should be similar to that of deletions. Although these trends are visible, caller-specific behaviours are more pronounced.
• Coverage above 30x resulted in marginal improvement in sensitivity for non-insertion events, while a reduction in coverage below 30x results in reduced sensitivity.
• Peak insertion sensitivity was not reached until at least 60x coverage.
• At 15x, the loss of sensitivity for DELLY, GRIDSS, and HYDRA was minimal indicating that these callers obtain peak sensitivity at a lower coverage that the others.

Varying the read length resulted in drastically different behavioural changes between callers (Error! Reference source not found.). For DP callers, apart from changes in insertion sensitivity, performance was relatively unaffected by read length until the reads were sufficiently long that the majority of fragments contained overlapping reads, at which point the sensitivity of DP callers plummeted. Sensitivity of both assembly and SR callers dropped dramatically on reads ≤50bp unless the caller also considered DPs.

Unlike SNV calling, the library fragment size distribution can have a marked effect on SV calling (Figure 3-3). For DP based callers, decreasing the fragment size enabled detection of smaller events but, as with varying the read length, when the median library fragment size approached less than twice the read length, sensitivity plummeted. SC based SR callers were unaffected by fragment size, and the effect on assembly depended on whether the assembler included read pairing information in the assembly process.
Figure 3-1 Effect of depth of coverage on variant caller performance
Figure 3-2 Effect of read length on variant caller performance
Figure 3-3 Effect of fragment length on variant caller performance
3.3.2 Caller-specific behaviour on simulated typical resequencing data

BreakDancer: Although the sensitivity of BreakDancer loosely matches that expected of a discordant read pair based caller, BreakDancer exhibits a high false discovery rate across all event types. For inversions and tandem duplications, the event size detection range is smaller than that of deletions. Reduced inversion performance could be explained by the presence of two underlying breakpoints, but with tandem duplications consisting of only a single breakpoint, and other DP callers not exhibiting this behaviour for either event type, this reduced detection capability cannot be explained on theoretical grounds.

Cortex: As a de novo assembly based caller, cortex should in theory be capable of detecting all events across all event sizes. Unfortunately, the results from cortex do not live up to the theoretical expectations. Across all events, the maximum sensitivity of cortex does not exceed 75%, the lowest of any of the callers evaluated. While the default maximum events of 64,000bp explains the dip in performance on large insertions and the lack of breakpoint detection capability, it does not explain the gradual reduction in deletion sensitivity, the extremely low sensitivity for insertion events, nor the lack of large tandem duplication calls. Whilst specificity is low for inversions and tandem duplications, cortex specificity on insertion and deletion events is good.

CREST: Similar to cortex, CREST does not reach 100% sensitivity, instead asymptoting at around 90% sensitivity. CREST does not detect insertions so no insertion calls are made and specificity is good except for inversions where false positive calls are made before true positives.

DELLY: Event size range detection is limited by the DP breakpoint identification stage. Insertions are not called. Sensitivity and specificity are good for events that can be detected. Curiously, DELLY can detect smaller inversions than it can deletions or tandem duplications.

GRIDSS: With the highest F-score for breakpoints and deletions, inversions, and tandem duplications larger than 50bp, GRIDSS performs well. The trough in insertion sensitivity around 64bp matches the cross-over point between SR and DP detection capability, as can be seen from the SOCRATES and BreakDancer insertion sensitivities. Overall ROC performance on inversions and tandem duplications is reduced due to a significantly higher FDR for small events (<50bp) than for large events.

HYDRA: HYDRA performance is similar to that of DELLY, albeit with lower specificity. Insertion and tandem duplication event size detection is worse than DELLY but the size of the minimum deletion detectable by HYDRA is less than half that of DELLY.
LUMPY: Although LUMPY’s minimum detectable event size is the worst of SR callers, it still outperforms DP-only callers and does so at a very low FDR. Notably, LUMPY made no false positive calls on either the deletion or breakpoint call sets. As with CREST, DELLY, and HYDRA, LUMPY does not support insertion calls.

manta: As a caller incorporating SR, DP, and assembly, manta performance is similar to that of GRIDSS even though their assembly approaches are quite different. Manta exhibits a smoother loss of sensitivity with insertion size as well as a longer maximum dateable insertion size. Unfortunately, this appears to come at the cost lower specificity across all event types, with the dip in tandem duplication sensitivity around 64bp considerably more pronounced in manta than in GRIDSS.

Pindel: Excluding large novel insertions, Pindel has the largest event size detection range of any of the callers evaluated and is the only caller that can reliably call events all the way down to 1bp in size. This sensitivity comes at the cost of a moderately high FDR and Pindel does not perform well when detecting arbitrary breakpoints. Curiously, Pindel exhibits a complete loss of sensitivity for 1kbp deletions as well as 2kbp tandem duplications.

SOCRATES: As a soft clipped based SR caller, the sensitivity of SOCRATES matches the theoretical expectations of a SC SR caller. As with GRIDSS, SOCRATES suffers from a high FDR for small inversion, and performs well on arbitrary breakpoint detection.

3.3.3 Caller-specific read length behaviour dependent on coverage

Of the DP-based callers, HYDRA most closely follows the expected behaviour of improved detection range with increased coverage. Unexpectedly, DELLY deletion and BreakDancer inversion displays the opposite trend with higher coverage resulting in a more abrupt detection range cut-off threshold. For SR-based callers, the detectable event size range does not change with coverage, although there is a preferential drop in sensitivity toward the edge of the detectable size range. As expected, GRIDSS, LUMPY, and manta all display a drop in sensitivity at the DP cut-off point.

Pindel displays unusual behaviour. Firstly, a lack of 1kbp deletion and 2kbp tandem duplication calls is present at all levels of coverage. Secondly, the abrupt change in sensitivity at 100bp indicates that Pindel treats events smaller than the read length differently to those larger than the read length. In a similar way, the drop in the sensitivity of manta for medium size deletion and tandem duplications only is not present for any other caller, nor is it readily explained by the published algorithm.
3.3.4 Caller-specific read length behaviour dependent on fragment size

To examine the effect of read length on caller performance, we varied read length from 36 to 250bp. Different read lengths resulted in drastically different behavioural changes between callers. For DP callers, apart from changes in insertion sensitivity, performance was relatively unaffected by read length until the reads were sufficiently long that the majority of fragments contained overlapping reads. For 2x250bp sequencing of 300bp fragments, all DP based callers performed poorly with maximum sensitivity ranging from a bit over 50% for DELLY, to around 2% for BreakDancer.

Assembly-based callers exhibited severely degraded performance on reads 50bp or shorter. In the cases of cortex and CREST, where assembly is required for variant calling, this resulted in an almost complete loss of variant calls for 36bp reads. For GRIDSS and manta, this loss was restricted to certain event types and sizes as variant calls could be made from SR and/or DP support even in the absence of assembly.

Similar to assembly-based callers, SR callers also exhibited a drastic drop in sensitivity for reads 50bp or shorter. This drop can be explained by the reduction in maximum soft clip length. Since aligners require a minimum read length for alignment, soft clips below this length cannot be re-aligned. An OEA SR caller, Pindel does not suffer this issue and is relatively unaffected by read length change, although again, Pindel failed to call 1kbp deletions and 2kbp tandem duplications under any conditions. Pindel large deletion and tandem duplication sensitivity above 1kbp was particularly poor for 2x250bp, suggesting that the Pindel algorithm has evolved from the published OEA-only algorithm and now uses discordant read pairs to seed the search locations of nearby OEA. Such an algorithm would allow Pindel’s missing calls to be plausibly explained by a bug in the event size algorithm choice logic of Pindel.

When considering events larger than 50bp, GRIDSS retained the best mean sensitivity and highest mean F score for all event types except insertions, for which cortex outperformed all over callers due to the underlying de novo assembly approach.

3.3.5 Caller-specific fragment size behaviour dependent on fragment size

Next, to assess the effect of library fragment size distribution, the mean was varied from 150bp to 500bp with a standard deviation of 10%. Compared to changes in coverage or read length, the effect of fragment size was more isolated (Figure 3-3).

Just as observed with increasing read length, decreasing fragment size and the occurrence of overlapping read pairs significantly reduce DP caller performance. Although this was less pronounced
at 2x100bp with 150bp fragments than at 2x250bp with 300bp fragments, the reduction in sensitivity was discernible across all DP callers. For the remaining non-overlapping fragment lengths, increasing the fragment size increased the minimum detectable event size as expected, except for deletions detected with DELLY, which retained the abrupt cut-off at 300bp.

As expected for a variant caller that does not incorporate any read pairing information, cortex was unaffected by fragment size with the exception of a small reduction in large insertion sensitivity for cortex. In contrast, although theoretically they should produce identical results for all fragment sizes, both CREST and SOCRATES results shows signs of fragment size dependence, with both unexpectedly exhibiting minor variations in sensitivity and a significantly increased false positive rate for 500bp fragments.

Pindel again showed signs for DP dependence for events over 1kbp, and a lack of detection capability of 1kbp deletions and 2kbp tandem duplications, except this time, Pindel was capable of detecting 2kbp tandem duplication but only for fragment sizes less than 300bp.

Of the remaining callers, GRIDSS proved to be the most robust to fragment size changes with the drop in sensitivity significantly less for GRIDSS than for both manta and lumpy. Curiously, GRIDSS exhibited the same increase in false discovery rate displayed by CREST and SOCRATES, but only for low confidence events not supported by GRIDSS assembly. For insertions, only GRIDSS and BreakDancer showed the improvement in maximum detectable event size expected from increasing fragment size.

When considering events larger than 50bp, GRIDSS retained the best mean sensitivity and highest F mean score for all event types except cortex on insertions, and manta on inversions which, even with lower sensitivity, exceeded the GRIDSS F score.

### 3.3.6 Excluded callers

Four structural variant callers were excluded from analysis for the following reasons:

CLEVER: On simulated data, CLEVER called many fewer results than expected (41 deletion calls, 0 for other events). Through personal correspondence with the software author it was determined that all published versions (1.1, 2.0rc1 and 2.0rc3) contained critical bugs causing either program failure, or call failure). As the recommended solution to compare against the most recent unstable, unreleased development version lacking both version and release information is not appropriate for reproducible evaluation, CLEVER results was excluded.

VariationHunter: No results for VariationHunter could be obtained. VariationHunter crashed with a “Segmentation fault (core dumped)” error on all simulations. VariationHunter could not be run on
NA12878 in a timely manner due to requirement of in excess of 20,000 hours of computation time required to run mrfast on the 50x whole genome sequencing data set.

SOAPsv: A script converting the user guide (10 pages of instructions containing 76 different steps) was not able to generated in the two days allotted. The number of steps and the requirements to recompile a new version of the software for every sample due to the presence of hard-coded file path in the source code indicates that SOAPsv was not designed for general purpose usage.

GASVPro: Results were significantly worse than expected for a read pair based caller. Upon investigation it was found that the LLR scoring used by GASV excessively favoured very large events. Combined with a filter removing overlapping events, most true positive GASV calls were filtered by the cluster pruning algorithm. In NA12878, a false positive deletion call from chr1:32,060,879 to chr1:243,114,737 with a LLR of $1.4 \times 10^{10}$ resulted in the removal of all chr1 deletion calls within this interval, thus removing most chr1 calls. Under these circumstances, it was decided that GASVPro results were not representative of read pair-based caller performance and results were excluded.

3.3.7 Human cell line data

To assess the actual performance of SV callers, we tested them on real sequencing data obtained from the NA12878 cell line. To eliminate any bias in the generation of a truth set, we used direct long read validation to assess variant calls with each variant requiring support from at least 3 long reads (see Methods for details). As with the simulated data, inexact matching was allowed. Only variants greater than 50bp in size were considered. Precision-recall curves were generated using either the variant score provided by the caller or, in the absence of a score, the total reads supporting the variant.

On 50x coverage 2x100bp PCR-free Illumina sequencing data, not only did all callers perform well below their maximum theoretical performance, but the callers varied considerably in both sensitivity and precision (Figure 3-4). In terms of specificity, GRIDSS stands out with a false discovery rate (FDR) almost half that of other callers, with LUMPY, CREST, and DELLY also performing well. When considering absolute sensitivity, HYDRA, GRIDSS and Pindel performed well. With the exception of HYDRA, the sensitivity of DP based callers was similar whilst all SR based callers varied considerably in both their FDR and sensitivity. Although cortex showed an initially low FDR, this dropped rapidly as sensitivity was less than ½ that of the other callers. In general, methods incorporating multiple signal sources outperformed single-source source approaches.
Figure 3-4 Variant caller performance on human cell line deletions

50x coverage of 2x100bp reads with 316bp mean fragment size of sample NA12878

For half of the variant callers, results could be separated in high and low confidence call sets based on whether a given variant call passed all filters applied by the variant caller. The effectiveness of the filtering varied considerably between variant callers. The filters applied by both SOCRATES and DELLY resulted in both reduced sensitivity and increased FDR, whereas the filters applied by cortex were particularly effective. For GRIDSS and manta, filtering reduced the FDR by 2-5%. Callers that reported a single variant calling set could be separated into those that only reported what appear to be high confidence variants (CREST, LUMPY), and those reporting all variants (BreakDancer, HYDRA, Pindel).

3.3.8 Performance, usability and reliability

Runtime performance of variant callers was measured for all simulated data sets (see Methods), with execution time and CPU usage varying by over two orders of magnitude (Figure 3-5). Excluding the CREST and Pindel outliers, single-source callers were the fastest, followed by multi-source callers, with the de novo assembly standing out as the most computationally expensive assembly approach. In cases such as the de novo assembly performed by cortex, poor relative runtime performance can be attributed to the computational expense of the underlying algorithm, but in the case of CREST, the
poor performance is due to the reliance on external tools. The use of CAP3 assembly combined with BLAT realignment of soft clips causes CREST assembly to run over an order of magnitude slower than either the Manta or GRIDSS assemblers.

Figure 3-5 Average variant caller runtime performance on simulated data sets

As is typical of many bioinformatics programs, usability and reliability varied considerably. Command-line execution of most programs was relatively straightforward, requiring execution of 1-2 commands, with an average of 7 command-line parameters (Table 4). Programs for which results were not able to be obtained generally required additional steps to be completed by the user (see Section 3.3.6). For example, SOAPsv has a detailed 10 page step-by-step guide containing 76 different steps, including editing the source code of programs to change hard-coded file paths. VariationHunter’s use of mrFAST as an aligner increased the command line complexity, but even after executing all steps as per the supplied manual, all simulations resulted in a Segmentation Fault crash. Other difficulties include CREST randomly hanging indefinitely (execution was halted after 180 days) and the requirement to start a BLAT server. The reliability of cortex was also problematic; when insufficient free memory was available to build a de Bruijn graph for a given kmer, graph construction would fail, an error message would be written to the log file, and execution would continue until completion. As a result, cortex
would output results based on only a subset of the requested kmers, with the only indication of failure
an error message partway through a large log file ending in a success message.

<table>
<thead>
<tr>
<th>Software</th>
<th>Commands</th>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>BreakDancer</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Cortex</td>
<td>2</td>
<td>23</td>
</tr>
<tr>
<td>CREST</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>DELLY</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>GRIDSS</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>HYDRA</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>LUMPY</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>manta</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Pindel</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>SOCRATES</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>VariationHunter/mrFAST</td>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>GASVPro</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>SOAPsv</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>CLEVER</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4 Variant caller command line usage requirements

3.4 Discussion

Variant detection is fundamentally important to sequencing analysis and the value of sensitive,
specific, usable, reliable, publicly available variant detection software is enormous. Through
comprehensive benchmarking of well-cited tools across a range of variant detection approaches, we
have shown that the tools available for general purpose SV calling vary widely across all of these traits.
Overall, several trends are visible across callers:

- A split read or assembly based method is required for single nucleotide precision calling.
- SR methods relying on soft-clipped reads perform poorly on ≤50bp.
- Incorporating multiple signals improves variant calling performance.
- Maximum sensitivity is reached between 15-30x coverage for most callers.
- \emph{De novo} assembly or specialised callers are needed to detect large novel insertions.
- DP callers perform poorly on events smaller than approximately 200bp.
- DP callers are sensitive to library fragment size distribution and are unable to call variants on
data sets with long read lengths, or small fragment sizes.

Unlike SR or assembly-based callers, DP callers have a trade-off in which long reads and shorter
fragments allow the detection of smaller events, at the cost of reduced signal strength. Critically, when
reads from the same fragment start overlapping once the fragment size is less than twice the read
length, the performance of DP based methods drops precipitously. This behaviour is most apparent in
BreakDancer in which moving from 2x100bp sequences to 2x250bp sequencing on the same library resulted in an increase of the FDR to 98% with a less than 10% change in the overall number of variant calls. Such behaviour is concerning as not only does this trend run contrary to the expectation that improved sequencing technology will result in better variant calling, but it does so abruptly, and with no obvious indication in the output call set.

Most SV callers are designed to detect only certain types of rearrangements. Deletion detection is ubiquitous amongst general purpose SV callers, but each other event type is only detectable by a subset of callers with many callers restricted to intra-chromosomal events due to software design. Callers such as GRIDSS, HYDRA, and SOCRATES address this by reporting breakpoints and leaving the interpretation to downstream analysis. BreakDancer, DELLY, and Pindel on the other hand, directly classify inversion events, but for complex events like chromothripsis, this results in false positive calls as these callers report an inversion event even when only one of the two breakpoints comprising the inversion are present.

Another concern with primarily read pair based callers is the lack of small event detection capability. The combination of a SNV/small indel caller such as samtools/bcfutils and a read pair caller will result in a call set that does not include events in the 50-200bp range. Detecting such events requires the use of an additional indel caller (e.g. SOAPindel [19]) with a wider detection range, or the use of an SV caller which utilises split reads or assembly as part of the variant calling algorithm.

Even for callers capable of detecting events of a given type, the approach taken to detection limits the range of detectable events. DP approaches are unable to detect small events. For insertions, SR approaches can only detect small events; DP approaches can detect events in a narrow size range; and the detection range of assembly-based approaches depends on whether break-end, breakpoint, or de novo assembly is used. These limitations can be overcome either by incorporating multiple detection approaches into the variant caller itself, or through the use of a consensus caller [57, 58] that generates a composite call set by combining the output of multiple callers with local breakpoint assembly validation.

That a caller is theoretically capable of detecting a given event does not mean that the event will be detected in practice. The false discovery rate on real data is well below that of simulated data for all callers and simulation results should be considered only as an upper bound on actual performance. Even on simulated data, BreakDancer, cortex, CREST, DELLY, and Pindel all exhibited unexpected behaviour for certain events. This makes SV caller selection difficult, especially when performance can differ dramatically for different event types and sizes and at different coverage, read lengths, and
library fragment size distributions. To aid in the selection of software, we recommend the following criteria be considered:

- Can the software detect events of the desired type in this data set?
- Can the software detect events in the desired size range in this data set?
- Are benchmarking results representative of this data set available?
- Can the software detect inter-chromosomal events (if required)?
- Does the software support any required multi-sample (population and/or somatic) analysis?
- Is a user guide or manual provided?
- Is a working example provided?
- Is the software actively maintained?
- Does the software have a test suite?
- Is the output format compatible with downstream tools?

Of the callers tested, CREST, DELLY, LUMPY and GRIDSS performed well, although CREST is slow. GRIDSS fulfils the above criteria for the widest range of data sets. Except at low coverage (≤15x), or on short reads (≤50bp), GRIDSS retains high sensitivity with the lowest FDR. For studies requiring high sensitivity with a relatively high FDR considered acceptable, GRIDSS, manta, HYDRA, and Pindel performed well. Pindel is notable for its reliable detection of events all the way down to 1bp.

For SV method developers, in addition to satisfying the above criteria, we recommend:

- Testing on simulated data across a range of read lengths, fragment size distributions, event sizes, and event types for which the tool is intended and documenting outcomes.
- Aborting with an error message if run on inappropriate data.
- Using standard file formats such as sam/bam/vcf/bedpe for inputs and outputs.

Chapter 4  Structural Variant Quality Score Calibrator

Whilst ideally a variant caller would report all events with no false positives, in practice this is not possible even for SNV callers. Downstream analysis of variant calls must rely on filtering of low quality variants to remove artefacts and reduce the false positive rate. How conservative the filtering is depends on the purpose of the analysis and how detrimental false positives are compared to false negatives. Whilst variant quality score calibration methods exist for SNVs and small indel callers, no such methods exist for structural variant callers. Most structural variant callers report only read counts, and the callers that do variant quality score do not report calibrated scores.

In this chapter I present an exploratory study into structural variant quality score calibration. This caller-agnostic approach calculated calibrated variant quality score based on caller annotation using either a reference truth set or long read sequencing of a representative sample. Although the quality scores reported are not well calibrated for all callers, this approach still represents a significant improvement over the arbitrary read count thresholds currently utilised for structural variant filtering.

4.1 Introduction

When performing variant analysis of genomic sequencing data, calibrated variant quality scores are immensely useful as they enable meaningful selection of the appropriate trade-off between sensitivity and specificity for the analysis. Whilst variant quality rescoring algorithms have been developed for single nucleotide variants and small indels [56], these techniques do not readily generalise to structural variants for two reasons. Firstly, calibrating based on a subset of known variants as, is performed by the GATK Variant Quality Score Recalibration algorithm [56], requires an accurate, comprehensive database of known variants such as dbSNP [9]. Although there are equivalent structural variant databases such as dbVar [10] for human germline variants, and COSMIC [151] for somatic variants, the fact that there are multiple orders of magnitude fewer structural variants than SNPs per sample results in an insufficiently large training set. Secondly, different structural variant events have wildly different signals and as such, must be treated differently. Unlike SNV calling in which a relatively simple model can be used to derive an accurate variant probability, accurately modelling the likelihood of a structural variant call being correct remains an open problem. For the most part, structural variant callers do not report quality scores and merely output supporting read counts. For those that do, uncalibrated quality scores are reported.

The VCF specifications [13] define the variant quality score as the Phred-scaled quality score for the assertion made in ALT. i.e. \(-10\log_{10} \text{prob}(\text{call in ALT is wrong})\) with ALT defined as the alternate (i.e. variant) allele. Unfortunately, this definition is not used in practice, and for variant callers which do
use this definition, the scores significantly overestimate the variant probability due to the model used. For example, the variant scoring model used by GRIDSS makes the assumption that the mapping quality score reported by the aligner accurately reflects the mismapping rate of the aligner and that mismapping by the aligner are distributed randomly. In practice, even when the former assumption is correct, the latter is not. All reads that can be uniquely mapped (that is, there exists a single mapping location in which the sequence edit distance between the read and the reference is minimised) are placed in the same optimal mapping location, thus resulting in a systematic bias which results in overconfident variant calls when read mapping is assumed to be independent.

Here, I introduce SVQSC, an approach for the calculating calibrated structural variant quality scores based on long reads sequencing, or other existing validated truth sets. To demonstrate the general applicability of the approach, I applied SVQSC on well-studied human cell line data on call sets from BreakDancer, CREST, DELLY and GRIDSS.

4.2 Methods

SVQSC takes as input a set of variant calls in the Variant Call Format (VCF) file format generated from a single variant caller. For callers not reporting calls in the VCF file format, a conversion script for each caller has been used to convert the custom file format to VCF version 4.1. Events 50bp or smaller as well as any VCF records not reporting any SVTYPE field are excluded and the remaining variants are broken down into their constituent breakpoint calls. Breakpoints are classified as deletion (DEL), insertion (INS), tandem duplication (DUP), inversion (INV), or inter-chromosomal (XCHR) based on the position and orientation of the two break-ends comprising the breakpoint, and the length of the inserted sequence. All breakpoints in which both break-ends do not occur on the same chromosome are considered inter-chromosomal events. Of the remaining breakpoints, those with both break-ends in the same orientation are considered inversions, those with breakend orientation indicative of repeated sequence as tandem duplications, and the remaining as deletions or insertions based on whether the length of any non-templated sequence inserted at the breakpoint is greater that the genome distance between the breakends.

SVQSC requires a training set with known truth. This training set can be derived by sequencing a sample with a known validated truth set, or by performing additional long read sequencing of a sample and using the long read sequencing to validate variant calls. Variant calls are considered true positive calls if the called variant matches a variant call in the truth set, or at least 5 long reads support the variant. Long reads are aligned using a split-read capable long read aligner such as bwa [4] and are considered to support a variant call if the read alignment contains an indel corresponding to the event, or the variant call is support by a split read alignment. To allow for matching of inexact variant calls,
when compared to a truth set event or long read, the variant call must overlap, be within 25% of the event size, and the break-end positions differ by less than 200bp. In the case of multiple matching variant calls, only the highest scoring variant call is considered as matching.

Calibrated quality scores are calculated using lasso regression on the quality score reported by the variant caller or by the reported supporting read count if no quality score is reported. The glmnet R package [152] is used to perform separate regressions for each breakpoint classification and the fitted probabilities are used to calculate the VCF Phred-scaled variant quality score. If fewer than 16 true positive variant calls, or 16 false positive variant calls, are present in the call set for any given breakpoint classification, that classification is considered to have insufficient data to train and no calibration is performed.

To investigate whether additional information provided by the variant caller can improve the quality of the variant call, models containing one or more additional inputs were generated based on the numerical annotations reported by that caller. Log transforms were taken of count-based fields and candidate models were generated for promising subsets based on the correlation of the variants with each other, and the output classification as given from generalized pairs plots [153] using the GGally R package.

SVQSV is implemented in R and is available at https://github.com/d-cameron/svqsc. Code for extracting structural variant break-end coordinates from a VCF file, data frame conversion, and breakpoint matching logic has been implemented in the StructuralVariantAnnotation R package available at https://github.com/PapenfussLab/StructuralVariantAnnotation. Conversion scripts for converting variant caller output to VCF are available at https://github.com/PapenfussLab/sv_benchmark.

4.3 Results

SVQSC models were trained on human cell line whole genome sequencing data from NA12878 and tested on HG002. Variant calls were called true or false positives based on their concordance with long read sequencing alignments. 2x101bp, PCR-free, 50x coverage data for NA12878 with median fragment length of 319bp was obtained from Illumina Platinum Genomics project (https://basespace.illumina.com/s/Tty7T2ppH3Tr ), and PacBio long reads were obtained from the 1000 Genomes Phase 3 SV group (http://ftp.1000genomes.ebi.ac.uk/ftp/phase3/integrated_sv_map/supporting/NA12878). For HG002, both 2x150bp, 60x coverage data for HG002 with median fragment length of 555, and PacBio validation data were obtained from the Genome In A Bottle Consortium (). As both the truth and
validation data sets are human germline, XCHR events were excluded from analysis. To evaluate the effectiveness of the models and calibration, precision-recall and decile-based calibration curves for each caller were produced.

For BreakDancer, the ReadCount model was derived from the num_Reads field, and the full model based on svlen, num_Reads, Size, and Score fields and a number of features can be observed from the BreakDancer results (Figure 4-1 BreakDancer Precision-recall and calibration curves). Most noticeable, are the lack of duplication calls, poor performance on inversion events, and reduced deletion sensitivity on the test data.

For CREST, the ReadCount model was derived from the sum of the left_softclipped_read_count and right_softclipped_read_count fields, and the full model based on svlen, left_assembly_length, left_softclipped_read_count, left_percent_identity, left_percent_multimapping, logleft_coverage,
right\textunderscore\text{assembly\_length}, \text{right\_softclipped\_read\_count}, \text{right\_percent\_identity}, \text{right\_percent\_multimapping}, \text{right\_coverage}, \text{with more consistent results than BreakDancer (Figure 4-2 CREST Precision-recall and calibration curves).}

For DELLY, the ReadCount model was derived from the sum of the PE and SR fields, and the full model based on svlen, PE, SR, MAPQ, and SRQ. DELLY results followed a similar trend to CREST, with reduced specificity on the training set and erratic duplication and inversion calibration results due to the small sample size (Figure 4-3 DELLY Precision-recall and calibration curves).
As the only caller reporting a variant quality score, the VCF QUAL field was used for the QUAL model with the full model based on svlen, the presence of assembly from one or both sides, reported size of exact and inexact homologies, and the SR, SRQ, RP, RPQ, and QUAL fields. As can be seen in the Figure 4-4 GRIDSS Precision-recall and calibration curves, the caller reported quality scores significantly overestimate the variant probabilities and are improved by calibration.
4.4 Discussion

Overall, SVQSC is able to provide meaningful variant quality scores based on either raw quality scores or read counts. Although this represents a significant improvement over the existing scoring approaches, there remain a number of outstanding issues with the calibrated values. Using the Hosmer-Lemeshow goodness of fit test, it can be shown that the fitted probabilities calculated by glmnet are not well calibrated for the training data. While the fits for the deletion calls are reasonable overall, the fits for duplication and inversion events are less so. This is in part due to different sizes of training sets, but also due the higher false discovery of these events compared to deletions.

Although the changes in read length, coverage, and fragment size of the data set were designed to test the robustness of the SVQSC training, analysis of the effectiveness of SVQSC is confounded by the marked difference in variant calling performance between the test and training data sets. As outlined
in Chapter 3, the performance of read pair based methods decreases as the read length increases. In the case of BreakDancer, the reduction in sensitivity can be attributed to the longer read lengths in the training data set and the precision-recall curves reflect this reduction. For DELLY and CREST, the increased false discovery rate on the test set compared to the training set cannot readily be explained. On simulated data, neither of these callers reduces in sensitivity when moving from 2x100bp to 2x150bp data, and longer fragment lengths should benefit DELLY. Although the HG002 test data was aligned using novoalign, and the NA12878 training data with bwa mem, such a difference should not affect the variant calling FDR so dramatically, especially given neither caller has a documented dependency on the aligner, and neither BreakDancer nor GRIDSS exhibits such a marked increase in FDR.

In addition to generating calibrated variant quality scores purely based on read count or caller-based quality scores, SVQSC can use the additional numerical attributes reported by the caller for each variant call to generate a more comprehensive model. In the case of BreakDancer, the more comprehensive model resulted in no change to the precision-recall performance, but for the remaining three callers, results were mixed. The FDR for tandem duplication calls was markedly reduced for DELLY and GRIDSS, but the expected FDR reduction in CREST was found only in the test set. In contrast, the expected 5% reduction in CREST deletion FDR relative to the read count model was obtained at all sensitivities except for a slightly elevated FDR for some high confidence calls. Deletion performance for DELLY was mixed, and for GRIDSS, the more comprehensive model performed worse than just using the quality score reported by GRIDSS. Similarly, the expected inversion FDR improvement for DELLY and GRIDSS was not present in the test set. Overall, the inconsistency of the performance improvements of the more comprehensive models do not justify their usage compared to simple calibration of the caller-reported quality scores/read counts for the level of heterogeneity present between the test and training data sets. It remains to be seen whether these models will produce an improved FDR and better calibrated scores for homogenous data sets such as those produced by the HiSeq X Ten systems.

In applications such as precision medicine, the primary focus is not on the calibration of entire call sets but on the identification of actionable variants with high confidence. Such variants are predominantly in and around genic sequence. Due to the low repeat content of these regions compared to intergenic sequence, they are easier for variant callers to identify. For these applications, an alternative approach that focuses on identifying and filtering known causes of false positive high confidence calls (such as sequence homology with pseudogenes) could produce a higher quality call set than de novo fitting of caller-reported variant attributes. More generally, these causes of false positives could be used in
addition to the caller-reporting attributes and a model generated that incorporates this additional information. Such information would improve on the hard filtering frequently used in downstream variant filtering. Hard filtering is appropriate for regions in which there are known issues with the reference genome, with filters such the ENCODE DAC and Duke blacklists, but is not appropriate for regions in which the false positive rate is elevated, but filtering would remove many true positives. Annotations of variant calls in regions such as those identified by RepeatMasker annotations as containing repetitive sequence, and well as variant-specific annotations such as the level of (both exact and inexact) homology between the breakpoint sequence and the reference sequences can be incorporated into the calibration model without requiring hard filtering.

4.5 Conclusion

Here I have presented, SVQSC, a model-based calibration method for generating VCF-compliant calibration quality scores for structural variant callers. SVQSC functions in the absence of an accurate, exhaustive structural variant mutation database, instead using sequencing of a representative sample, in conjunction with either a known truth set, or long read sequencing to train a calibrated structural variant scoring model. Signal heterogeneity between different types of structural variants is handled by modelling each type independently. Due the training data requirements, SVQSC is primarily of use in large cohort studies, in which the cost of long read sequencing can be amortized over the project cost, or for which sequencing data from well-studied samples for the same sequencing platform can be obtained.

By generating empirically calibrated variant quality scores for multiple callers, SVQSC can assist in the generation of ensemble call sets. Much like the choice of variant filters to apply to a given call set, the optimal annotations to use to train the SVQSC calibration model remains an open question.

Although the results presented are not well calibrated in general, it is expected that further work will improve on these results. The use of an alternative model such as Gaussian mixture model (as is done by the GATK Variant Quality Score Recalibration algorithm), or a neural network is likely to outperform the linear model used in SVQSC, especially since the many of the regression variables have disjoint distributions. Breakpoint sequence context annotations including RepeatMasker repeat class and level of sequence homology between the variant and reference alleles have been successfully used when performing downstream filtering of GRIDSS and can be generalised into SVQSC. Finally, further investigation into the causes of false positives calls for each variant caller is likely to reveal additional attributes that can be included in the SVQSC model to improve classification performance.
Chapter 5  Conclusion

This final chapter is divided into four sections. The first section outlines additional applications of GRIDSS not covered in Chapter 2. These applications demonstrate the broad utility GRIDSS in both a research and a clinical setting. The second outlines several enhancements that can be made to GRIDSS either to improve variant calling performance through the incorporation of additional methods, or to widen the scope of GRIDSS to handle additional sequencing platforms and library preparation techniques. The third section speculates on the future of genomic rearrangement detection in light of recent algorithmic and technological advances. Finally, this chapter concludes with a summary of the contributions this research has made to the accurate and specific detection of genomic rearrangements in short read sequencing data.

5.1 Applications

Although useful as a general purpose structural variant caller, the low false discovery rate of GRIDSS compared to existing state-of-the-art approaches makes it particularly well suited to precision medicine and clinical genomics. As outlined in Chapter 2, GRIDSS has been used for the identification of patient-specific tumour biomarkers for use in relapse monitoring [154], improving chromothriptic breakpoint detection, and malaria vaccine candidate validation in the AT-rich genome of Plasmodium falciparum.

In additional to the above studies, GRIDSS has been employed to investigate the structural evolution of cancer using sequencing data obtained as part of the CAncer tISsue Collection After Death (CASCADE) programme [155]. This rapid autopsy program enrols patients presenting with advanced stage disease across several tumour streams and provides a unique opportunity to understand the driving molecular changes by comparing the molecular changes between visceral metastatic tumour deposits against matched primaries and regional metastasis for the same patient. Using whole genome sequencing from both patient autopsy as well as patient-derived xenografts, GRIDSS has been able to reveal not only the gains, but also the losses of complex genomic rearrangements occurring during tumour evolution.

In the clinical cancer setting, work is currently underway on incorporating GRIDSS structural variant calling into a Molecular Genomics Diagnostic Reporting Package (https://github.com/PapenfussLab/PathOS). PathOS enables bioinformatics filtering, classification of variants, annotation, curation and clinical reporting all in an ISO15189 accredited framework [156, 157]. A large part of the clinical sequencing performed by PathOS uses exome or targeted capture panel sequencing. For the targeted capture panel, exon capture of cancer-associated genes is
augmented by intron tiling of known cancer fusion genes. Promising preliminary results indicate that GRIDSS is not only able to identify fusions between genes with tiled introns, but it can also identify gene fusions between a tiled gene and a non-tiled gene. For actionable fusion genes, such as ALK, this capability has direct clinical relevance.

5.2 Further work

Whilst combining genome-wide break-end assembly with read pair and split read analysis, as is done by GRIDSS, outperforms existing structural variant calling approaches, it is by no means the final stage in the evolution of genomic rearrangement detection. New and improved analysis techniques are continually being developed and indeed, there currently exist callers with more sophisticated approaches to split read identification or discordant read pair analysis which could be adopted by GRIDSS. The introduction of new sequencing technology as well as improvement in existing sequencing technologies require constant re-evaluation of detection approaches as the optimal approach for today’s data is not necessarily suitable for the data sets of tomorrow. As such, there are a large number of major enhancements that could be utilised to further improve the variant calling of GRIDSS.

5.2.1 Read depth

Fundamentally, GRIDSS is a genomic breakpoint detector. Although useful to downstream analysis, no effort is made to translate breakpoints into events. This is a result of an intentional design decision to implement GRIDSS as a modular software suite in which new components may be incorporated, and existing components replaced with alternative implementations. Whilst not directly useful to breakpoint identification, read depth information is extremely useful for event classification. A deletion-like breakpoint with no change of coverage across the breakpoint is unlikely to be a simple deletion event. Conversely, no coverage change is expected for inversions. Similar expectations apply to all simple events. Whilst most callers do not incorporate read depth information (Table 1), GASVPro[32], LUMPY[69], PeSV-Fisher[66], and SV-Bay[67] do. Except for GASVPro, read depth analysis could be deferred to an event classification stage after breakpoint identification. In the case of GASVPro, read depth information is incorporated into the MCMC assignment of multi-mapping reads to variant calls, an approach not possible if read depth information is incorporated after the multi-mapping reads have already been assigned to variant call. To ascertain how this limitation of the GRIDSS architecture impacts variant calling capability, a detailed comparison between the multi-mapping read allocation approaches taken by GRIDSS and GASVPro would be required. Unfortunately, the runtime of GASVPro (in excess of 28 days for a single 50x WGS human sample) makes such as comparison impractical.
In summary, although GRIDSS does not incorporate read depth information, this is not required for breakpoint detection. Although the incorporation of read depth information does improve breakpoint detection confidence, this information can be performed downstream of the breakpoint identification in event classification software such as CLOVE (https://github.com/PapenfussLab/clove).

5.2.2 Improved read pairing analysis

Like most paired-end callers, GRIDSS takes a relatively simplistic approach to the identification of discordant read pairs by considering all reads pairs with inferred fragment lengths falling outside a fixed interval defined by the library fragment size distribution. Such an approach has poor performance in the <100bp range as most of the read pairs spanning the variant fall within the interval bounds and thus are considered concordant. CLEVER [33] improves on this by considering all read pairs and identifying clusters of reads inconsistent with the library fragment size distribution. This approach enables detection of insertions and deletions in the 20-100bp range. Such an approach could be incorporated into GRIDSS. As GRIDSS has the option to process discordant read pairs based on the “proper pair” SAM flag, a CLEVER-like program that replaced the aligner-assigned value of this flag would be all that would be required. Such an approach would allow independent reuse by other tools and be consistent with the modular architecture of GRIDSS.

5.2.3 Genotyping

Although GRIDSS performs variant detection, it does not yet perform genotyping. For highly rearranged somatic genomes, traditional genotyping is of limited value as not only can the ploidy of such genomes can differ wildly from the progenitor germline, but a single SV can be homozygous at one breakend, and heterozygous at the other. For simple events in less rearranged genomes, such as germline indels, duplications, and inversions, genotyping is of significant value to many studies. As GRIDSS already reports counts for reference-supporting read, reference-supporting read pairs, variant-supporting split read, and variant-supporting read pairs, extending GRIDSS to also perform genotyping would be a relatively straight-forward enhancement.

5.2.4 Large novel insertion detection

Detection of large insertions is performed using one of two approaches. The first, and most common, is to augment the reference genome with the sequence of interest (such as is done in VirusSeq, VirusFinder, and ViralFusionSeq [92-94]). The second approach, which is required if the inserted sequence is unknown, is to directly identify the insertion of novel sequence. Although GRIDSS does perform break-end assembly, it does not attempt to match breakend assemblies containing novel sequence, thus the detection capability is limited to insertions smaller than the fragment size. At present, only NovelSeq[62], or full de novo assembly are capable of detecting novel insertions larger
than the fragment size. Theoretically, by extending GRIDSS to merge the contigs resulting from a traditional de novo assembly of unmapped reads with the unpaired break-ends, GRIDSS would be able to outperform both pure de novo assembly and NovelSeq in large insertion detection. As demonstrated in Chapter 2, positional de Bruijn graph assembly is less vulnerable to kmer repeats than traditional de Bruijn graph assembler thus improving the assembly quality of the novel sequence adjacent to the insertion site. Although this is the same high-level approach taken by NovelSeq, GRIDSS should outperform NovelSeq through a combination of positional de Bruijn graph assembly, and inclusion of soft clipped and split read into the break-end assembly.

Further demonstrating the value of modularised software, an alternative to reimplementing components of the NovelSeq pipeline into GRIDSS, the clustering and local assembly used by NovelSeq to identify putative break-end could be replaced wholesale by the GRIDSS genome-wide break-end assembler.

5.2.5 Mate pair integration

Although most Illumina short read sequencing is performed on single end or paired end libraries, mate pair library preparation (https://www.illumina.com/technology/next-generation-sequencing/mate-pair-sequencing_assay.html) allows for short read sequencing of the ends of DNA fragments several to tens of kilobases long. These long-insertion paired-end DNA libraries are useful in a number of applications including de novo assembly, genome finishing, structural variant detection and the identification of complex genomic rearrangements. These libraries differ from typically paired-end libraries not only in size the originating DNA, but also in the orientation of the reads themselves.

As demonstrated in Chapter 3, increasing the length of the DNA spanned by a pair of reads linearly increases the strength of the read pair signal.

To incorporate mate pair libraries into GRIDSS two key enhancements would be required. Firstly, the orientation of read pairs would need to be determined by the library distribution itself. Secondly, as mate pair libraries contain a non-trivial amount of ‘contamination’ of paired-end reads, the expected placement of discordant read pairs would need to allow for both traditional paired-end placement as well as mate pair placement for mate pair libraries.

5.2.6 Chromium 10X genomics integration

The Chromium system from 10X genomics is rapidly gaining popularity in a number of areas, one of which is structural variant detection. The Chromium system is a microfluidics based molecular barcoding library preparation system that enables the barcoding of DNA fragments suitable for short read sequencing from originating DNA fragments hundreds of kilobases in length. With each
microfluidic droplet containing as few as 5 DNA molecules, these barcodes can be used to associate reads sequenced on a short read sequencer hundreds of kilobases apart, thus providing an alternative to long read sequencing. The extreme length of these “linked-reads” enables phasing, de novo assembly, and structural variant calling with an accuracy not possible with other technologies. Early structural variant calling results [158] indicate that, even when only considering linked-reads associations, variant calling significantly outperforms calling from short read sequencing alone. Linked-read information can be used to enhance GRIDSS in multiple ways. Most obviously, linked-reads can be used directly in the variant calling process itself. Additionally, a modified form of the alignment-constrained assembly used by GRIDSS can be used in place of a traditional de Bruijn graph assembler for the assembly of linked-read clusters at lower coverage than would otherwise be possible.

5.2.7 Long read integration

Long read sequencing technology such as the Pacific Biosciences Single Molecule Real-Time (SMRT) sequencing technology and Oxford Nanopore technologies (ONT) produce sequences reads over an order of magnitude longer than short reads sequencers. Such reads are invaluable for resolving structural variants within repetitive regions as, due to the lack of long range information in short read sequencing, unambiguous placement of variants within large perfect repeats is not possible for any short read variant caller. Recent work using read sequencing on haploid human cell lines indicate that the majority of structural variation within humans remains undetected by typical short read sequencing pipelines [159]. Whether genome-wide break-end assembly outperforms existing SV calling approaches on long read sequencing data remains an open question.

5.2.8 Overlap-Layout-Consensus genome-wide break-end assembler

Typically, short read sequencers (such as the Illumina sequencing platform) have a relatively low error rate with errors dominated by substitution errors. Such reads are amenable to kmer based de Bruijn graph assembly as sequencing errors typically result in small bubbles in the assembly graph which can be error corrected relatively easily. In contrast, reads from SMRT and ONT sequencers have a high error rate with small insertions and deletions dominating the error profile. Such reads do not readily assemble using any kmer based de Bruijn graph approach. Even with the relatively short kmer length of 25 used by GRIDSS, the probably of two reads originating from the same genomic loci sharing kmers is relative low. With such an error profile, the majority of such an assembly graph would consist of kmers containing errors thus precluding bubble-popping as a viable error correction strategy. Indel errors are particular problematic for positional de Bruijn graph assembly. Unlike traditional de Bruijn graph assembly in which an indel will result in a bubble containing paths of different lengths, an indel
error will cause a positional de Bruijn graph to branch. Since each kmer is constrained to a single position, such branches will only ever re-join the trunk if an additional compensating indel also occurs.

As positional de Bruijn graph assembly is not viable for data from long read sequencers, an alternate assembler would be required. For long read data, genome-wide break-end assembly would require an Overlap-Layout-Consensus (OLC) assembler that, in addition to the constraints generated through read alignment overlap, further constrains the read alignment overlap based on the alignments of the reads relative to the reference. That is, only ‘nearby’ reads would be assembled together. Due to the significant reduction in read overlaps required to be considered, such an approach would be significantly faster than that traditional OLC assembly. Given the increasingly long reads obtained by long read technologies, it remains to be seen whether such an approach provides any improvement over existing windowed local assembly approaches on long read data.

5.2.9 RNA-Seq

The genome-wide break-end assembly approach used by GRIDSS has a number of applications outside of genomic rearrangement detection from DNA-Seq data including the closely related problem of RNA-Seq gene fusion detection. Although the same fundamental techniques of split read alignment and discordant read pair analysis are used for both problems, there is little overlap in the tools used. In part, this can be explained as a duplication of effort by researchers applying the same techniques in different application domains, but details of the application of the technique differ sufficiently that a tool designed for one type of data will perform poorly on the other type. Since RNA-Seq sequences expressed transcripts instead of the genomic DNA directly, aligning reads produced by RNA-Seq will result in reads aligning to exons and across exon-exon junctions. Given the median human exon length is 136bp (as defined by UCSC hg19 gene annotations), the majority of RNA-Seq fragments will span multiple exons, thus confounding any transcript naïve approach to discordant read pair analysis. The abundance of exon-exon junctions in RNA-Seq has resulted in a distinct ecosystem of RNA-Seq aligners specifically designed for improved mapping of junction-spanning reads [160]. Similarly plentiful are gene fusion detectors [161] with those including an assembly step amongst the best performing [162]. Such results indicate that genome-wide break-end assembly has potential for improving RNA-Seq gene fusion calls.

In its existing form, the GRIDS genome-wide break-end assembler is not suitable for RNA-Seq analysis as for each connect non-reference assembly sub-graph, only the highest weight path is reported³. This

³ Technically, GRIDSS can be used for RNA-Seq analysis if the reads were aligned to the set of transcripts instead of the reference genome using a multi-mapping aligner with a priori knowledge of the library insert size distribution. Such an approach would not be compatible with typical RNA-Seq pipelines, and would still only
results in the loss of assembly contigs corresponding to more lowly expressed transcripts. For example, consider the break-end assembly at exon A in which both A-B-C and A-B-D transcripts are expressed and exon B is short. The assembly subgraph will contain the exon B sequence followed by branches to C and D. In this scenario, the assembler will report B-C or B-D but not both. Extending the assembler to report all break-end contigs would enable genome-wide break-end assembly on RNA-Seq data.

5.3 Future of genomic rearrangement detection

Several recent technological and algorithmic advances have already started making their mark in the landscape of genomic rearrangement detection and are likely to become increasingly important in the coming years. Most prominent are long read sequencing technologies such as SMRT and ONT. Although variable in length, long reads enable much more comprehensive characterisation of structural variants. Experiments conducted on the more developed PacBio SMRT technology have revealed the short read sequencing performed for the 1000 Genomes Project did not detected the majority of structural variants present [159]. Although a less mature technology, early results for ONT include the resolution of the previously inaccessible human X chromosome CT47-repeat region using reads up to 42kb long [163].

An alternative to long read sequencing is the generation of synthetic long reads from short read sequencing. Of particular note is the 10X Genomics Chromium system that is well suited to single cell sequencing, phasing, and structural variant calling. By barcoding sets of short reads originating from the same long fragments with the same barcode long range structural information can be added to short reads. As it only requires an additional library preparation step, this approach provides a low cost, immediately available alternative to true long read sequencing. Early results indicate that unlike standard short read sequencing, complex structural rearrangements containing breakpoint spaced greater than the library fragment size (typically ~500bp) with could be resolved [158]. Synthetic long read technologies do have their limitations and their detection capability suffers more from the presence of repetitive sequence than true long read technologies. Unlike true long read sequencing, synthetic long reads are unable to unambiguously place all constituent short read in the presence of repeat-rich sequence. Such limitations do not entirely preclude the use of synthetic long read variant calling in repetitive regions and there is ample scope for the development of improved synthetic long read variant calling techniques over the coming years.

assemble the dominant fusion transcript in promiscuous breakpoint situations where an exon is involved in two different fusion events.
Another trend in structural variant has been the development of ensemble calling tools. Ensemble callers call variants not from the reads directly, but by combining the result of multiple variant callers, typically augmented by a targeted breakpoint assembly validation step [57, 58, 164]. By combining multiple general-purpose variant callers with specialised callers (such as RetroSeq [90] for transposon detection), a call set can be generated with greater sensitivity than any constituent caller. The challenge with ensemble calling is controlling the false discovery rate. Recent ensemble callers such as Parliament [71] are capable of integrating calls from multiple callers across multiple technologies. Such callers are likely to become increasingly prevalent as a means to overcome the limitations of any individual variant caller as, even for callers relying on the same signal and using the same algorithm, the resultant call sets are not the same.

The most significant recent algorithmic advance in structural variant detection has been the development of genome graphs. In contrast to the current approach of using a single canonical linear reference genome sequence, genome graphs encode the sequence diversity of the population directly within the graph itself. By explicitly representing a genome using a mathematical graph structure, not only can the allelic bias towards to reference allele be eliminated, but the read mapping rate can be improved. This has such a profound effect on variant calling that even a simplistic variant calling procedure on a genome graph performs comparably to a state-of-the-art method on the linear human reference genome [165]. The nascent ecosystem of genome graph read mapping and variant calling tools is still immature and, for structural variant detection, there exists an opportunity to further improve on the state-of-the-art approaches by adapting the sophisticated methods developed for linear reference genomes to genome graphs.

5.4 Conclusion

Accurate and specific detection of genomic rearrangements remains an open problem in computational biology. Identifying the breakpoints associated with genomic rearrangements from short read sequencing is typically performed through split read analysis, discordant read pair analysis, assembly, or a combination of these approaches. Of the assembly approaches used, de novo assembly is the only approach that detects large novel insertions, but it is the most vulnerable to repetitive sequence. Breakpoint assembly can only be used as a validation step with variant calling performed prior to assembly. Targeted and windowed break-end assembly can be used prior to variant calling but, as with breakpoint assembly, are vulnerable to repetitive sequence within the window/targeted region. This thesis contributes to the resolution of the genomic rearrangement detection problem by introducing a novel form of assembly: genome-wide break-end assembly. By encoding read alignments as positional constraints with the assembly graph itself, break-end assembly can not only...
be performed without any targeting or windowing, but can be made more robust to the presence of repetitive sequence. This robustness enables reliable assembly at lower coverage using shorter kmers.

The genome-wide break-end assembly has been implemented using a positional de Bruijn graph. This graph encodes not only sequence kmers, but also the expected position of these kmers into the graph itself. Previously used for error correction of assembly contigs, this thesis demonstrates that although the raw assembly graph is an order of magnitude larger than a *de novo* assembly graph, through extensive optimisation and data streaming, positional de Bruijn graph assembly is possible at a genome scale in a small memory footprint.

This assembly approach was combined with split read and read pair analysis using a probabilistic variant calling model in the Genome Rearrangement IDentification Software Suite (GRIDSS). Through benchmarking on both real and synthetic data sets, I have shown that not only does genome-wide break-end assembly outperform windowed break-end assembly, but it is through the incorporation of break-end assembly contigs into the variant calling process itself that GRIDSS is able to outperform alternative approaches.

GRIDSS has been compared to existing state-of-the-art methods in the most comprehensive benchmarking of genomic rearrangement detection software to date. It has been shown that once the 15x coverage required for robust assembly is reached, GRIDSS outperforms other methods across a wide range of sequencing depths, library fragment sizes, and read length.

In conclusion, the incorporation of genome-wide break-end assembly into a probabilistic variant calling model allows genomic rearrangement breakpoints to be identified with half the false discovery rate than can be achieved otherwise and represents a significant advance in the quest for accurate and specific detection of genomic rearrangements.
Chapter 6 Bibliography


### Appendix A: Gene Fusions detected by INTEGRATE and GRIDSS

<table>
<thead>
<tr>
<th>gene</th>
<th>fusion_junction</th>
<th>validated</th>
<th>has validation probe</th>
<th>NUMRNAseq</th>
<th>NUMMSseq</th>
<th>INTEGRATE breakpoints (63x coverage)</th>
<th>GRIDSS breakpoints (21x coverage)</th>
<th>confidence</th>
<th>Qualified status</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIF3K</td>
<td>CYP39A1</td>
<td>19</td>
<td>39123318</td>
<td>6</td>
<td>46607405</td>
<td>6 39124623</td>
<td>6 39124623</td>
<td>High</td>
<td>chr19</td>
</tr>
<tr>
<td>MDH1</td>
<td>RP11-357C3</td>
<td>2</td>
<td>63826426</td>
<td>x</td>
<td>62619783</td>
<td>6 3827838</td>
<td>X 62622236</td>
<td>High</td>
<td>chr2</td>
</tr>
<tr>
<td>LMF1</td>
<td>FOXX2</td>
<td>16</td>
<td>1020788</td>
<td>7</td>
<td>80559179</td>
<td>1 1012777</td>
<td>1 80553042</td>
<td>High</td>
<td>chr16</td>
</tr>
<tr>
<td>GS1-12AK5.2</td>
<td>RP11-147C23.1</td>
<td>7</td>
<td>65915058</td>
<td>1</td>
<td>96471749</td>
<td>1 65898846</td>
<td>1 96461779</td>
<td>High</td>
<td>chr7</td>
</tr>
<tr>
<td>RNTG</td>
<td>RP1F1</td>
<td>6</td>
<td>89559474</td>
<td>1</td>
<td>84946693</td>
<td>1 89554518</td>
<td>1 84911888</td>
<td>High</td>
<td>chr6</td>
</tr>
<tr>
<td>RP11-83A24.2</td>
<td>KCNQ5</td>
<td>4</td>
<td>140374839</td>
<td>6</td>
<td>79300296</td>
<td>4 1.4E+08</td>
<td>6 73890696</td>
<td>High</td>
<td>chr4</td>
</tr>
<tr>
<td>PLA2R1</td>
<td>RBMS1</td>
<td>2</td>
<td>160832579</td>
<td>2</td>
<td>161159989</td>
<td>2 1.6E+08</td>
<td>2 16116017</td>
<td>High</td>
<td>chr2</td>
</tr>
<tr>
<td>RA87A</td>
<td>LRCH3</td>
<td>3</td>
<td>128445202</td>
<td>3</td>
<td>197592294</td>
<td>3 1.2E+08</td>
<td>3 197591082</td>
<td>High</td>
<td>chr3</td>
</tr>
<tr>
<td>SHISA5</td>
<td>RP11-24C3.2</td>
<td>3</td>
<td>48520856</td>
<td>3</td>
<td>48482722</td>
<td>3 48514370</td>
<td>3 4843272</td>
<td>High</td>
<td>chr4</td>
</tr>
<tr>
<td>UNCSC</td>
<td>STPG2</td>
<td>4</td>
<td>96199410</td>
<td>4</td>
<td>98162249</td>
<td>4 96179911</td>
<td>4 98364655</td>
<td>High</td>
<td>chr5</td>
</tr>
<tr>
<td>TMEM104</td>
<td>RAB37</td>
<td>17</td>
<td>77291717</td>
<td>17</td>
<td>77252395</td>
<td>17 7729728</td>
<td>17 7727126</td>
<td>High</td>
<td>chr6</td>
</tr>
<tr>
<td>MAV5</td>
<td>PANK2</td>
<td>20</td>
<td>38434060</td>
<td>20</td>
<td>3888573</td>
<td>20 3863395</td>
<td>20 3887764</td>
<td>High</td>
<td>chr7</td>
</tr>
<tr>
<td>ZNF378</td>
<td>CEP164P1</td>
<td>10</td>
<td>43046005</td>
<td>10</td>
<td>46516970</td>
<td>10 4302411</td>
<td>10 45582412</td>
<td>High</td>
<td>chr8</td>
</tr>
<tr>
<td>POUP2F1</td>
<td>NME7</td>
<td>1</td>
<td>167382351</td>
<td>1</td>
<td>169102055</td>
<td>1 1.6E+08</td>
<td>1 16912992</td>
<td>High</td>
<td>chr9</td>
</tr>
<tr>
<td>NRL</td>
<td>DHRS5-A14</td>
<td>14</td>
<td>24583931</td>
<td>14</td>
<td>24410367</td>
<td>14 24572404</td>
<td>14 24402426</td>
<td>High</td>
<td>chr10</td>
</tr>
<tr>
<td>KPN1A</td>
<td>EFSEC</td>
<td>13</td>
<td>122168421</td>
<td>13</td>
<td>12796567</td>
<td>13 1.2E+08</td>
<td>13 127905568</td>
<td>High</td>
<td>chr11</td>
</tr>
<tr>
<td>NOM1</td>
<td>MNX1-A52</td>
<td>7</td>
<td>156756720</td>
<td>7</td>
<td>15679975</td>
<td>7 1.5E+08</td>
<td>7 156774864</td>
<td>High</td>
<td>chr12</td>
</tr>
<tr>
<td>EIFTD2</td>
<td>KIF18B</td>
<td>17</td>
<td>42942298</td>
<td>17</td>
<td>43006392</td>
<td>17 42941595</td>
<td>17 43008040</td>
<td>High</td>
<td>chr13</td>
</tr>
<tr>
<td>OSBPL9</td>
<td>RP11-191G24.1</td>
<td>1</td>
<td>52179751</td>
<td>1</td>
<td>52026554</td>
<td>1 52180174</td>
<td>1 51992695</td>
<td>High</td>
<td>chr14</td>
</tr>
<tr>
<td>MFSD3</td>
<td>MROH1</td>
<td>8</td>
<td>145736286</td>
<td>8</td>
<td>145254025</td>
<td>8 1.4E+08</td>
<td>8 145248358</td>
<td>High</td>
<td>chr15</td>
</tr>
<tr>
<td>WARS</td>
<td>DEGS2</td>
<td>14</td>
<td>100082691</td>
<td>14</td>
<td>100060147</td>
<td>14 1.0E+08</td>
<td>14 100062851</td>
<td>Low</td>
<td>chr16</td>
</tr>
<tr>
<td>HNRNPU2L</td>
<td>AHNK</td>
<td>11</td>
<td>62484461</td>
<td>11</td>
<td>62201363</td>
<td>11 62484336</td>
<td>11 62315471</td>
<td>High</td>
<td>chr17</td>
</tr>
<tr>
<td>CCDCl17</td>
<td>CTA-29E10.6</td>
<td>22</td>
<td>29179633</td>
<td>22</td>
<td>29179600</td>
<td>22 29207341</td>
<td>22 29207801</td>
<td>High</td>
<td>chr18</td>
</tr>
<tr>
<td>LDLRAD3</td>
<td>PRR5L</td>
<td>11</td>
<td>36120011</td>
<td>11</td>
<td>36422457</td>
<td>11 36150547</td>
<td>11 36344953</td>
<td>High</td>
<td>chr19</td>
</tr>
<tr>
<td>CASD1</td>
<td>GNGL1</td>
<td>7</td>
<td>94183887</td>
<td>7</td>
<td>9355403</td>
<td>7 94231522</td>
<td>7 93552377</td>
<td>High</td>
<td>chr20</td>
</tr>
<tr>
<td>RP11-85K15.2</td>
<td>BAZ2A1</td>
<td>14</td>
<td>35390795</td>
<td>14</td>
<td>35348868</td>
<td>14 35368100</td>
<td>14 35382660</td>
<td>High</td>
<td>chr21</td>
</tr>
<tr>
<td>POMZP3</td>
<td>ZP3</td>
<td>7</td>
<td>76241039</td>
<td>7</td>
<td>76089756</td>
<td>7 76043545</td>
<td>7 76246161</td>
<td>Low</td>
<td>chr22</td>
</tr>
<tr>
<td>PPP1R10</td>
<td>ABCF1</td>
<td>6</td>
<td>30573812</td>
<td>6</td>
<td>30539337</td>
<td>6 30573812</td>
<td>x 30573950</td>
<td>Low</td>
<td>chr23</td>
</tr>
<tr>
<td>BIRC6</td>
<td>LTBP1</td>
<td>2</td>
<td>32631567</td>
<td>2</td>
<td>33246273</td>
<td>2 32631567</td>
<td>2 33246273</td>
<td>Low</td>
<td>chr24</td>
</tr>
<tr>
<td>TBC1D16</td>
<td>ACC012354.6</td>
<td>17</td>
<td>77983959</td>
<td>17</td>
<td>45193557</td>
<td>17 77983959</td>
<td>17 45193557</td>
<td>Low</td>
<td>chr25</td>
</tr>
<tr>
<td>FOSL2</td>
<td>BRE</td>
<td>2</td>
<td>28616689</td>
<td>2</td>
<td>28460609</td>
<td>2 28616689</td>
<td>2 28460609</td>
<td>Low</td>
<td>chr26</td>
</tr>
<tr>
<td>MANBA</td>
<td>UBE2D3</td>
<td>10</td>
<td>10368187</td>
<td>10</td>
<td>103731012</td>
<td>10 103731012</td>
<td>10 103731012</td>
<td>Low</td>
<td>chr27</td>
</tr>
<tr>
<td>SMDT1</td>
<td>NDUF6-A51</td>
<td>22</td>
<td>42475958</td>
<td>22</td>
<td>42520052</td>
<td>22 42475958</td>
<td>22 42520052</td>
<td>Low</td>
<td>chr28</td>
</tr>
</tbody>
</table>

**High** indicates high confidence, **Low** indicates low confidence, **Not called by G** indicates that the fusion was not called by GRIDSS.
<table>
<thead>
<tr>
<th>Gene</th>
<th>Description</th>
<th>Log2 Fold Change</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSE1</td>
<td></td>
<td>16 85391249</td>
<td>1 1 11 43</td>
</tr>
<tr>
<td>CNPY3</td>
<td></td>
<td>6 42903408</td>
<td>6 42929950 1 1 9 21</td>
</tr>
<tr>
<td>CNPY3</td>
<td></td>
<td>6 42905945</td>
<td>6 42911536 1 1 8 23</td>
</tr>
<tr>
<td>GNA12</td>
<td></td>
<td>7 2883487</td>
<td>7 2998265 1 1 7 24</td>
</tr>
<tr>
<td>RBM14</td>
<td></td>
<td>11 66384528</td>
<td>11 66526514 1 1 7 5</td>
</tr>
<tr>
<td>FARS2</td>
<td></td>
<td>6 5613553</td>
<td>6 5852296 1 1 5 7</td>
</tr>
<tr>
<td>FSD1L</td>
<td></td>
<td>9 108234286</td>
<td>9 108200562 1 1 5 18</td>
</tr>
<tr>
<td>SIK1</td>
<td></td>
<td>21 44841518</td>
<td>21 44473757 1 1 4 11</td>
</tr>
<tr>
<td>PALLD</td>
<td></td>
<td>4 16958951</td>
<td>13 45009019 1 1 9 45</td>
</tr>
<tr>
<td>PKD1P5</td>
<td></td>
<td>16 18483571</td>
<td>19 2229784 1 1 7 6</td>
</tr>
<tr>
<td>EIF4A2</td>
<td></td>
<td>3 18650537</td>
<td>2 153551136 1 1 6 16</td>
</tr>
<tr>
<td>DHX36</td>
<td></td>
<td>3 153994536</td>
<td>3 31199090 1 1 6 18</td>
</tr>
<tr>
<td>POLR2J3</td>
<td></td>
<td>7 102207444</td>
<td>X 54954100 1 1 5 4</td>
</tr>
<tr>
<td>TRMT10B</td>
<td></td>
<td>9 37762114</td>
<td>7 97481780 1 1 5 17</td>
</tr>
<tr>
<td>FBXL8</td>
<td></td>
<td>16 67195840</td>
<td>2 70278437 1 1 5 1</td>
</tr>
<tr>
<td>NABP1</td>
<td></td>
<td>2 192543854</td>
<td>8 19694563 1 1 5 5</td>
</tr>
<tr>
<td>MBNL2</td>
<td></td>
<td>13 9784785</td>
<td>1 225755116 1 1 5 27</td>
</tr>
<tr>
<td>FOSL2</td>
<td></td>
<td>2 28616689</td>
<td>3 100091566 1 1 4 5</td>
</tr>
<tr>
<td>TPT1-AS1</td>
<td></td>
<td>13 45944317</td>
<td>4 48517292 1 1 4 2</td>
</tr>
<tr>
<td>HNRNPR</td>
<td></td>
<td>1 23660011</td>
<td>17 66039021 1 1 4 6</td>
</tr>
<tr>
<td>L2IC</td>
<td></td>
<td>1 9996577</td>
<td>7 6431555 1 1 4 3</td>
</tr>
<tr>
<td>PCNL2</td>
<td></td>
<td>1 233398704</td>
<td>2 10580994 1 1 4 9</td>
</tr>
<tr>
<td>RP11-19F5S.4</td>
<td></td>
<td>14 61438990</td>
<td>1 26798997 1 1 4 22</td>
</tr>
<tr>
<td>RPL4</td>
<td></td>
<td>15 66795396</td>
<td>12 53691634 1 1 4 4</td>
</tr>
<tr>
<td>BCL2L2-</td>
<td></td>
<td>14 23793498</td>
<td>8 99054078 1 1 4 4</td>
</tr>
<tr>
<td>DAP3</td>
<td></td>
<td>1 15568691</td>
<td>X 154292308 1 1 4 13</td>
</tr>
<tr>
<td>ANP32B</td>
<td></td>
<td>9 100773671</td>
<td>12 6879558 1 1 3 4</td>
</tr>
<tr>
<td>ATP2A2</td>
<td></td>
<td>12 110734542</td>
<td>20 3430035 1 1 3 2</td>
</tr>
<tr>
<td>BCL2L1</td>
<td></td>
<td>2 111878765</td>
<td>16 25175919 1 1 3 1</td>
</tr>
<tr>
<td>CLEC6</td>
<td></td>
<td>1 11867247</td>
<td>16 5495129 1 1 3 2</td>
</tr>
<tr>
<td>CPSF6</td>
<td></td>
<td>12 69656342</td>
<td>5 36183942 1 1 3 16</td>
</tr>
<tr>
<td>NKX2</td>
<td></td>
<td>3 42674315</td>
<td>6 131943119 1 1 3 7</td>
</tr>
<tr>
<td>RNGT</td>
<td></td>
<td>6 89559474</td>
<td>13 46544600 1 1 3 3</td>
</tr>
<tr>
<td>SOS2</td>
<td></td>
<td>14 50647290</td>
<td>15 90229777 1 1 3 11</td>
</tr>
<tr>
<td>UQCRCL2</td>
<td></td>
<td>16 21987564</td>
<td>5 40828053 1 1 3 3</td>
</tr>
<tr>
<td>CTSF</td>
<td></td>
<td>20 4452679</td>
<td>10 12133626 1 1 3 9</td>
</tr>
<tr>
<td>DUSP16</td>
<td></td>
<td>12 12653453</td>
<td>17 79851428 1 1 3 3</td>
</tr>
<tr>
<td>EXOSC1</td>
<td></td>
<td>9 133574781</td>
<td>19 9939268 1 1 3 2</td>
</tr>
<tr>
<td>HIST1H2AC</td>
<td></td>
<td>6 26124977</td>
<td>3 134039762 1 1 3 11</td>
</tr>
<tr>
<td>IARS</td>
<td></td>
<td>9 95004008</td>
<td>14 21960455 1 1 3 3</td>
</tr>
<tr>
<td>LEMD1</td>
<td></td>
<td>1 205389583</td>
<td>8 99054078 1 1 3 6</td>
</tr>
<tr>
<td>MAPK9</td>
<td></td>
<td>5 179675973</td>
<td>17 80364358 1 1 3 2</td>
</tr>
<tr>
<td>PCBD2</td>
<td></td>
<td>5 134246156</td>
<td>2 10747437 1 1 3 13</td>
</tr>
<tr>
<td>Gene</td>
<td>Symbol</td>
<td>Chromosome</td>
<td>Band</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>------------</td>
<td>------</td>
</tr>
<tr>
<td>PCMTD1</td>
<td>POM121C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PNISR</td>
<td>KLHDC1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAB10</td>
<td>MX1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRPA</td>
<td>VPRBP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TJP1</td>
<td>PLOD3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOP2A</td>
<td>GLRX5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZFR</td>
<td>ZMYM5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMM1</td>
<td>RP11-390P2.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FND3C8</td>
<td>RBM39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IQGAP1</td>
<td>RPL30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAA15</td>
<td>PLK1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEK7</td>
<td>CMC2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNF145</td>
<td>PLOD2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOX2</td>
<td>POLR1D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZNF521</td>
<td>RAB28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AGPAT5</td>
<td>CTD-2541M15.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERO1A</td>
<td>FERM2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD109</td>
<td>EEF1A1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUCO</td>
<td>TBX19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZC3H8</td>
<td>SUCLG1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZNF37BP</td>
<td>C10orf25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLC30A9</td>
<td>OC1201</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADK</td>
<td>VCL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO99403.2</td>
<td>POMZP3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RP11-282O18.3</td>
<td>CDK2AP1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAV1</td>
<td>TES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLIP4</td>
<td>FAM179A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMADHC</td>
<td>EPC2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POM121</td>
<td>VKORC1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RP11-123O10.4</td>
<td>GRIP1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPF1</td>
<td>PRKACB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST3GAL1</td>
<td>SLA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCDC122</td>
<td>ENOX1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EIF4H</td>
<td>GPNMB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTBP2</td>
<td>BUB3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARSJ</td>
<td>MTHFD2L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST7</td>
<td>TES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOP2B</td>
<td>NGLY1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZNF519</td>
<td>RAB31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRD1</td>
<td>RP6-109B7.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLC39A11</td>
<td>NSF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZHX1</td>
<td>MRPL13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANXA2</td>
<td>NARG2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DNTTIP2  GCLM  1  94344633  1  94360284  1  1  3  1
FAM208A  PSMD6  3  56672734  3  63999283  1  1  3  4
GGCT  GARS  7  30544185  7  30665850  1  1  3  7
GNAO1  RP11-46309.5  16  56370772  16  86598924  1  1  3  27
NOP56  SNRPB  20  2637541  20  2443407  1  1  3  8
TBC1D9  RNF130  5  179315103  5  179390564  1  1  3  3
AH1  PSMB1  6  135748305  6  170858201  1  1  2  5
DLX5  NHLRC3  7  96653581  13  39613273  1  1  6  12
RG55  MTHFD1L  1  163276023  6  151413603  1  1  6  8
KDM2A  ZNF532  11  66994311  18  56601665  1  1  5  12
DPP9  ENDOG  19  4714093  9  131583018  1  1  4  7
TRIM52  NUP133  5  180687002  1  229634053  1  1  4  29
AFG3L2  RNPC3  18  12351305  18  104097656  1  1  3  12
TNFRSF1A  PTMS  12  6450942  12  6878769  1  1  3  9
BCL11B  PPP2R5C  14  99723808  14  102323023  1  1  2  3
GBE1  RYBP  3  81810526  3  72428578  1  1  2  13
NCOA7  CTA-293F17.1  6  126107359  7  20368479  1  1  5  3
TRAF2  ITPR1  9  139811067  3  4878458  1  1  5  15
RTFDC1  RP4-199D16.1  20  55052180  1  25370327  1  1  5  28
ACTR6  DHDDS  12  100594697  1  26759382  1  1  2  9
ERBB2IP  KTN1  5  65222634  14  56078737  1  1  2  12
FBXO32  ANXA13  8  124525438  8  124724993  1  1  4  4
ACO26150.8  GOLGA6B  15  30838172  15  30898046  0  0
LINCO0472  RP11-15A1D6.1  6  72126113  6  71943360  0  0
RPS15A  RPLP1  16  18794278  15  69745158  0  0
ACO21224.1  HNRNPA1  18  29993199  12  54680737  0  0
TAF15  HNRNPA2B1  17  34171732  7  26233314  0  0
PRKAR1A  H3F3A  17  66522053  1  226259052  0  0
PPHLN1  NCK1  12  42749024  3  136667101  0  0
ALG13  SNRPA1  X  110925522  15  101827215  0  0
ANP32A  EEF1B2  15  69075295  2  207026762  0  0
DHR51  AURKA  14  24759809  20  54967352  0  0
NUTF2  PATL1  16  67899132  11  59405884  0  0
SNHG8  PDS5B  4  119200292  13  33268360  0  0
DDX46  RSN11-D5S1  5  134121276  7  77313568  0  0
INS5  TPST1  9  103035388  7  65821812  0  0
RBM14-RBM4  INT510  11  66407594  8  19709160  0  0
ZSCAN31  AGTPBP1  6  28321639  9  88236200  0  0
BOLA2B  RP11-347C12.2  16  30204679  16  30300106  0  0
CSF4  MYH16  7  99054994  7  98836534  0  0
LINCO0623  SEC22B  1  14439564  1  145109524  0  0
CAB39  SP100  2  231577945  2  231359131  0  0
ACACA  AATF  17  35536201  17  35376304  0  0
ATP6V0E1  ERGIC1  5  172410967  5  172359439  0  0
ITGB6  RBMS1  2  160998493  2  161143595  0  0
<table>
<thead>
<tr>
<th>Gene</th>
<th>Symbol</th>
<th>Chromosome</th>
<th>Start</th>
<th>End</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLN6</td>
<td>AAGAB</td>
<td>15</td>
<td>68510874</td>
<td>67495236</td>
<td>0 0</td>
</tr>
<tr>
<td>PAK6</td>
<td>BUB1B</td>
<td>15</td>
<td>40532882</td>
<td>40457254</td>
<td>0 0</td>
</tr>
<tr>
<td>HIF1A</td>
<td>PRKCH</td>
<td>14</td>
<td>62162557</td>
<td>61857943</td>
<td>0 0</td>
</tr>
<tr>
<td>LUZP1</td>
<td>SRRM1</td>
<td>1</td>
<td>23435515</td>
<td>24975350</td>
<td>0 0</td>
</tr>
<tr>
<td>SLC39A8</td>
<td>PAP5S1</td>
<td>4</td>
<td>103225474</td>
<td>108615162</td>
<td>0 0</td>
</tr>
<tr>
<td>MED15</td>
<td>FTH1</td>
<td>22</td>
<td>20905723</td>
<td>61735103</td>
<td>0 0</td>
</tr>
<tr>
<td>BTF3</td>
<td>ENSG000000219298</td>
<td>5</td>
<td>72794272</td>
<td>150299052</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-33B1.1</td>
<td>SEP7P2</td>
<td>4</td>
<td>120382872</td>
<td>45798673</td>
<td>0 0</td>
</tr>
<tr>
<td>LGALS1</td>
<td>PPP2CA</td>
<td>22</td>
<td>38075813</td>
<td>133561245</td>
<td>0 0</td>
</tr>
<tr>
<td>RPS10</td>
<td>RP11-46D20.2</td>
<td>6</td>
<td>34393809</td>
<td>60594128</td>
<td>0 0</td>
</tr>
<tr>
<td>AP0D06571.2</td>
<td>FAM157C</td>
<td>7</td>
<td>12856170</td>
<td>90203008</td>
<td>0 0</td>
</tr>
<tr>
<td>STK17A</td>
<td>YTHDC1</td>
<td>6</td>
<td>43664117</td>
<td>69197821</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-384K6.6</td>
<td>LINC00265</td>
<td>4</td>
<td>119549191</td>
<td>39822398</td>
<td>0 0</td>
</tr>
<tr>
<td>NSUN2</td>
<td>PRSS23</td>
<td>5</td>
<td>6599353</td>
<td>86518673</td>
<td>0 0</td>
</tr>
<tr>
<td>AC137934.1</td>
<td>AC002055.4</td>
<td>16</td>
<td>90288794</td>
<td>51223721</td>
<td>0 0</td>
</tr>
<tr>
<td>APP</td>
<td>RPL10</td>
<td>21</td>
<td>27423509</td>
<td>153626718</td>
<td>0 0</td>
</tr>
<tr>
<td>SNHG7</td>
<td>AC07183.4</td>
<td>9</td>
<td>139619507</td>
<td>84568948</td>
<td>0 0</td>
</tr>
<tr>
<td>ACLY</td>
<td>UBA1</td>
<td>17</td>
<td>40034356</td>
<td>47073726</td>
<td>0 0</td>
</tr>
<tr>
<td>EVA1C</td>
<td>FAM6863P</td>
<td>21</td>
<td>33080549</td>
<td>8099484</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-87H9.2</td>
<td>FRG1B</td>
<td>9</td>
<td>69082110</td>
<td>29614327</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-34P13.7</td>
<td>AC093642.5</td>
<td>1</td>
<td>129055</td>
<td>243080935</td>
<td>0 0</td>
</tr>
<tr>
<td>XX-C2158C6.2</td>
<td>AC093642.5</td>
<td>6</td>
<td>171054606</td>
<td>243081039</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-14N7.2</td>
<td>AC027612.6</td>
<td>1</td>
<td>148933291</td>
<td>91843394</td>
<td>0 0</td>
</tr>
<tr>
<td>WASH3P</td>
<td>ABC7-42389800N19.1</td>
<td>15</td>
<td>102513423</td>
<td>87703</td>
<td>0 0</td>
</tr>
<tr>
<td>WASH3P</td>
<td>XXal-YRM2039.2</td>
<td>15</td>
<td>102513795</td>
<td>17719</td>
<td>0 0</td>
</tr>
<tr>
<td>WASH6P</td>
<td>ABC7-42389800N19.1</td>
<td>15</td>
<td>155252677</td>
<td>88017</td>
<td>0 0</td>
</tr>
<tr>
<td>MTERFD2</td>
<td>RP11-597A11.1</td>
<td>2</td>
<td>242033847</td>
<td>20128737</td>
<td>0 0</td>
</tr>
<tr>
<td>RPS10</td>
<td>RP11-425110.1</td>
<td>6</td>
<td>34393809</td>
<td>46450700</td>
<td>0 0</td>
</tr>
<tr>
<td>RP11-592N21.1</td>
<td>RLPL7</td>
<td>15</td>
<td>71634012</td>
<td>47017815</td>
<td>0 0</td>
</tr>
<tr>
<td>RPS29</td>
<td>RP11-156G14.6</td>
<td>14</td>
<td>50053094</td>
<td>35592783</td>
<td>0 0</td>
</tr>
<tr>
<td>ZCCHC7</td>
<td>RCC1</td>
<td>9</td>
<td>37305541</td>
<td>28835417</td>
<td>0 0</td>
</tr>
<tr>
<td>PELP1</td>
<td>DNAJC2</td>
<td>17</td>
<td>45754079</td>
<td>102963078</td>
<td>0 0</td>
</tr>
<tr>
<td>PTP4A2</td>
<td>TNFRSF21</td>
<td>1</td>
<td>32372022</td>
<td>47200730</td>
<td>0 0</td>
</tr>
<tr>
<td>PSM4A</td>
<td>MLF1</td>
<td>20</td>
<td>60711791</td>
<td>158317803</td>
<td>0 0</td>
</tr>
<tr>
<td>STRC</td>
<td>E2F4</td>
<td>15</td>
<td>43910857</td>
<td>67229685</td>
<td>0 0</td>
</tr>
<tr>
<td>MTHFD1L</td>
<td>RPL32</td>
<td>6</td>
<td>151413722</td>
<td>12881839</td>
<td>0 0</td>
</tr>
<tr>
<td>RPS10</td>
<td>RP3-375P9.2</td>
<td>6</td>
<td>34393809</td>
<td>34711953</td>
<td>0 0</td>
</tr>
<tr>
<td>U91319.1</td>
<td>SHISA9</td>
<td>16</td>
<td>13444037</td>
<td>13297406</td>
<td>0 0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>RP11-1212A22.1</td>
<td>16</td>
<td>16431463</td>
<td>18428708</td>
<td>0 0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>RP11-958N24.1</td>
<td>16</td>
<td>16431463</td>
<td>15029254</td>
<td>0 0</td>
</tr>
<tr>
<td>CTD-2334D19.1</td>
<td>PRR16</td>
<td>5</td>
<td>120125649</td>
<td>119800340</td>
<td>0 0</td>
</tr>
<tr>
<td>BUD31</td>
<td>ATP5J2-PTCD1</td>
<td>7</td>
<td>99017239</td>
<td>99026827</td>
<td>0 0</td>
</tr>
<tr>
<td>PDXDC2P</td>
<td>PKD1P1</td>
<td>16</td>
<td>70016253</td>
<td>18455134</td>
<td>0 0</td>
</tr>
<tr>
<td>Gene</td>
<td>Chromosome</td>
<td>Start</td>
<td>End</td>
<td>Log2 Fold Change</td>
<td>FDR</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>----------</td>
<td>----------</td>
<td>------------------</td>
<td>-------</td>
</tr>
<tr>
<td>TAS2R14</td>
<td>12</td>
<td>11126254</td>
<td>12</td>
<td>11083261</td>
<td>0</td>
</tr>
<tr>
<td>PRH2</td>
<td>15</td>
<td>20724201</td>
<td>15</td>
<td>28501235</td>
<td>0</td>
</tr>
<tr>
<td>HERC2</td>
<td>17</td>
<td>16721553</td>
<td>17</td>
<td>39739392</td>
<td>0</td>
</tr>
<tr>
<td>ACO22596.6</td>
<td>16</td>
<td>16434163</td>
<td>16</td>
<td>15219548</td>
<td>0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>16</td>
<td>51065352</td>
<td>10</td>
<td>95173536</td>
<td>0</td>
</tr>
<tr>
<td>CAST</td>
<td>17</td>
<td>39739392</td>
<td>17</td>
<td>99711837</td>
<td>0</td>
</tr>
<tr>
<td>BMS1</td>
<td>8</td>
<td>89048442</td>
<td>8</td>
<td>99057316</td>
<td>0</td>
</tr>
<tr>
<td>MATN2</td>
<td>10</td>
<td>81664805</td>
<td>10</td>
<td>81664805</td>
<td>0</td>
</tr>
<tr>
<td>AC022596.6</td>
<td>9</td>
<td>97101118</td>
<td>9</td>
<td>97101118</td>
<td>0</td>
</tr>
<tr>
<td>KRT14</td>
<td>10</td>
<td>51462115</td>
<td>16</td>
<td>15204678</td>
<td>0</td>
</tr>
<tr>
<td>RP11-307E17.8</td>
<td>17</td>
<td>62897188</td>
<td>17</td>
<td>44376892</td>
<td>0</td>
</tr>
<tr>
<td>HERC2</td>
<td>17</td>
<td>44053192</td>
<td>7</td>
<td>74366193</td>
<td>0</td>
</tr>
<tr>
<td>AC159540.1</td>
<td>17</td>
<td>51314980</td>
<td>10</td>
<td>51087876</td>
<td>0</td>
</tr>
<tr>
<td>LINC00342</td>
<td>17</td>
<td>10704245</td>
<td>2</td>
<td>109379693</td>
<td>0</td>
</tr>
<tr>
<td>AC012309.5</td>
<td>17</td>
<td>99927805</td>
<td>7</td>
<td>72511686</td>
<td>0</td>
</tr>
<tr>
<td>DTX2P1</td>
<td>17</td>
<td>76648314</td>
<td>7</td>
<td>99918707</td>
<td>0</td>
</tr>
<tr>
<td>ARO12309.5</td>
<td>17</td>
<td>76758920</td>
<td>5</td>
<td>155145104</td>
<td>0</td>
</tr>
<tr>
<td>LINC00665</td>
<td>17</td>
<td>38982422</td>
<td>19</td>
<td>36804224</td>
<td>0</td>
</tr>
<tr>
<td>KRT14</td>
<td>17</td>
<td>81162874</td>
<td>6</td>
<td>80982938</td>
<td>0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>17</td>
<td>70154364</td>
<td>16</td>
<td>74401963</td>
<td>0</td>
</tr>
<tr>
<td>RP11-307E17.8</td>
<td>17</td>
<td>43635773</td>
<td>17</td>
<td>44376892</td>
<td>0</td>
</tr>
<tr>
<td>WDR41</td>
<td>17</td>
<td>37748296</td>
<td>19</td>
<td>36804224</td>
<td>0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>17</td>
<td>18462115</td>
<td>16</td>
<td>15204678</td>
<td>0</td>
</tr>
<tr>
<td>POL2R1J4</td>
<td>17</td>
<td>62897188</td>
<td>17</td>
<td>44376892</td>
<td>0</td>
</tr>
<tr>
<td>PMS2P5</td>
<td>17</td>
<td>44053192</td>
<td>7</td>
<td>74366193</td>
<td>0</td>
</tr>
<tr>
<td>PARG</td>
<td>17</td>
<td>51314980</td>
<td>10</td>
<td>51087876</td>
<td>0</td>
</tr>
<tr>
<td>RGD3</td>
<td>17</td>
<td>13265992</td>
<td>7</td>
<td>132990631</td>
<td>0</td>
</tr>
<tr>
<td>CHCD3</td>
<td>17</td>
<td>43635773</td>
<td>17</td>
<td>44376892</td>
<td>0</td>
</tr>
<tr>
<td>RGD3</td>
<td>17</td>
<td>76758920</td>
<td>5</td>
<td>155145104</td>
<td>0</td>
</tr>
<tr>
<td>WDR41</td>
<td>17</td>
<td>37748296</td>
<td>19</td>
<td>36804224</td>
<td>0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>17</td>
<td>76648314</td>
<td>7</td>
<td>99918707</td>
<td>0</td>
</tr>
<tr>
<td>RGD3</td>
<td>17</td>
<td>81162874</td>
<td>6</td>
<td>80982938</td>
<td>0</td>
</tr>
<tr>
<td>CHCD3</td>
<td>17</td>
<td>13265992</td>
<td>7</td>
<td>132990631</td>
<td>0</td>
</tr>
<tr>
<td>RGD3</td>
<td>17</td>
<td>76758920</td>
<td>5</td>
<td>155145104</td>
<td>0</td>
</tr>
<tr>
<td>WDR41</td>
<td>17</td>
<td>37748296</td>
<td>19</td>
<td>36804224</td>
<td>0</td>
</tr>
<tr>
<td>PKD1P1</td>
<td>17</td>
<td>76648314</td>
<td>7</td>
<td>99918707</td>
<td>0</td>
</tr>
<tr>
<td>RGD3</td>
<td>17</td>
<td>81162874</td>
<td>6</td>
<td>80982938</td>
<td>0</td>
</tr>
</tbody>
</table>
Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:
Cameron, Daniel Lee

Title:
Improving the detection of genomic rearrangements in short read sequencing data

Date:
2017

Persistent Link:
http://hdl.handle.net/11343/191281

File Description:
Improving the detection of genomic rearrangements in short read sequencing data

Terms and Conditions:
Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.