Abstract of "Algorithms for Identifying Structural Variants in Human Genomes" by Anna Ritz, Ph.D., Brown University, May 2013

Variation in genomes occurs in many forms, from single nucleotide changes to gains and losses of entire chromosomes. Large-scale rearrangements, called structural variants (SVs), are associated with numerous diseases and are common in cancer genomes. However, many SVs in mammalian genomes are found in highly repetitive regions, complicating their detection and characterization. Ongoing development of genomic technologies invite new algorithmic approaches to SV detection.

In this thesis, we present a collection of four algorithms that identify SVs using data from current and emerging genomic technologies. The first algorithm is designed for a technology called array-comparative genomic hybridization (aCGH), which measures the number of copies of DNA segments present in a test genome relative to a known reference genome. aCGH data is useful for measuring deletions and duplications, and it is obtainable for thousands of individuals from a single population or disease group. We describe a method to identify SVs that are common to a group of individuals, and apply the method to aCGH data from cancer patients. We recover an SV in prostate cancer that is known to be biologically important, and we infer a number of novel SVs in brain cancer.

Our other algorithms are designed for DNA sequencing technologies, which measure a broader range of SVs than aCGH data with the tradeoff of higher cost. One DNA sequencing technology, strobe sequencing, yields multiple sequences from a single, contiguous fragment of DNA. While strobes provide longer sequenced portions of DNA compared to other sequencing technologies, the per-base error rate is substantially higher. Our algorithms for SV detection exploit the benefits of multiply-linked DNA sequences while being robust to high sequencing error rates. We describe the first published method for SV detection using strobe sequencing, which finds the smallest number of SVs (relative to a known reference genome) that explain the strobes. We then improve upon our method with a probabilistic algorithm that better models the strobe sequencing data. Finally, we describe a *de novo* assembly algorithm for strobe sequencing data when a reference genome is unavailable. We assess the performance of these algorithms on simulated and real strobe sequencing data, and conclude that with appropriate algorithms, strobe sequencing compares favorably to other DNA sequencing technologies.

Algorithms for Identifying Structural Variants in Human Genomes

by

Anna Ritz

A dissertation submitted in partial fulfillment of the
requirements for the Degree of Doctor of Philosophy
in the Department of Computer Science at Brown University

Providence, Rhode Island

May 2013

This dissertation by Anna Ritz is accepted in its present form by
the Department of Computer Science as satisfying the dissertation requirement
for the degree of Doctor of Philosophy.

Date ______________ ______________________________
Benjamin J. Raphael, Director

Recommended to the Graduate Council

Date ______________ ______________________________
Casey Dunn, Reader

Date ______________ ______________________________
David Laidlaw, Reader

Date ______________ ______________________________
Eli Upfal, Reader

Approved by the Graduate Council

Date ______________ ______________________________
Peter M. Weber
Dean of the Graduate School

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Variation in genomes, including single nucleotide changes and deleted, duplicated, and rearranged segments of DNA, has played key roles in genetic disorders and diseases. Recently, there have been efforts to accumulate growing amounts of data for genetic variation studies, including the International HapMap Project [33, 34] and the 1,000 Genomes Project [32] for identifying variation in healthy individuals and The Cancer Genome Atlas [9, 85, 94] for identifying variation in individuals with cancer. These datasets have motivated the algorithms community to develop new models and methods for identifying genetic variation.

It was initially believed that single nucleotide mutations (termed single nucleotide polymorphisms) were the main contributing factor to genetic variation [34]. However, there has been growing appreciation for the effect of larger-scale rearrangements [128, 130, 153]. These rearrangements, termed *structural variants* (SVs) or *structural aberrations*, include segment insertions, deletions, inversions, and translocations (Figure 1.1). There are more total base pairs in the human genome affected by SVs than single nucleotide polymorphisms [123]. The Database of Genomic Variants [161] currently lists 101,923 SVs. Although some of these variants are redundant and/or erroneous, it is clear that structural variation is an important component of human genome variation.

SVs have been associated with various diseases including Autism [84], Parkinson's [25], and Schizophrenia [149], as well as a host of cancers [2, 18, 30]. Cancer in particular has been called a "disease of alterations" [85], where a series of genetic mutations occur within a patient's lifetime. Identifying SVs in disease groups is important to gain understanding of the underlying biological mechanisms with which the disease progresses.

Current experimental methods now allow for the detection of SVs whose sizes range from tens of bases (such as diallelic insertions and deletions [152]) to entire chromosomes (such as duplicated chromosomes [44]). This proposal focuses on identifying SVs that range from a few hundred base pairs to entire chromosomal gains and losses. Genomic technologies that measure SVs are rapidly being developed and improved, and analyses that were impractical when this thesis work began are now a reality. We have taken

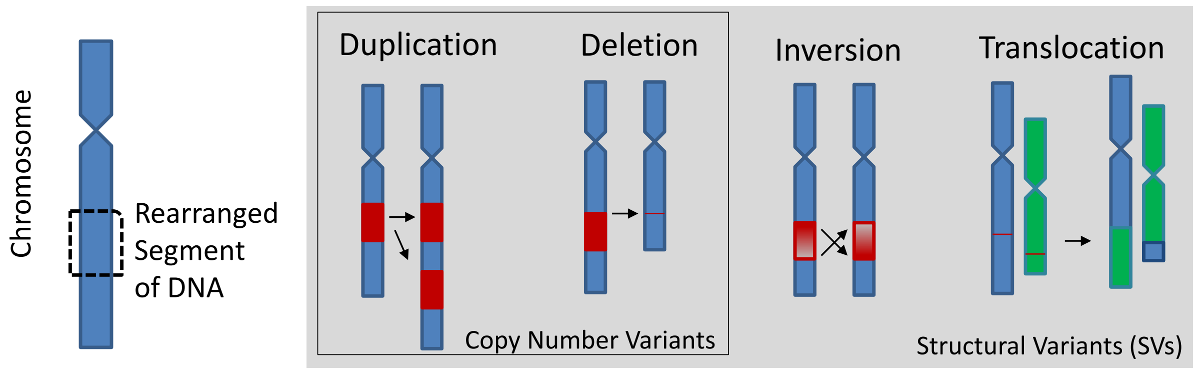

Figure 1.1: **Structural Variants.** Structural variants are large segments of DNA that have been rearranged relative to some reference genome. Duplications involve the addition of a DNA segment, deletions involve the removal of a DNA segment, inversions involve flipping a DNA segment, and translocations involve swapping DNA segments from different chromosomes. Copy number variants are the SVs that alter the number of copies of a DNA segment, such as duplications and deletions.

a technology-driven approach to identifying SVs that explicitly leverage the contributions of various technologies while making them robust to the technological limitations. We have developed a collection of four algorithms whose development have been driven by two different genomic technologies.

This dissertation proceeds as follows: Chapter 1 provides a general background of the two genomic technologies and a summary of research contributions from the four algorithms. Chapter 2 presents work for the detection of somatic copy number variants (a subset of SVs) from multiple individuals with the same type of cancer using an array-based technology. We then describe an emerging DNA sequencing technology (termed strobe sequencing by Pacific Biosciences), and present methods for SV detection using this technology in Chapters 3 and 4. Finally, Chapter 5 presents a *de novo* assembly method for strobe sequencing data.

1.1 Recurrent Copy Number Variant Detection in Cancer Genomes

Copy number variants (CNVs) are a subset of SVs that result in a different number of copies of a genomic segment, and include deletions, amplifications, and unbalanced translocations (Figure 1.1). While CNVs do not encompass all types of variants (they do not include balanced rearrangements such as inversions), they are detectable from measurements of the number of copies of DNA fragments in an individual genome.

Cancer genomes are highly-rearranged due to the *somatic* mutations that occur within a patient's lifetime. Some of the resulting SVs might not affect cancer progression, and identifying the important SVs (the *driver* SVs) in cancer is often difficult. Driver SVs that affect cancer progression tend to appear in multiple individuals with the same cancer; thus, we wish to find *recurrent* SVs that appear in many patients. Recurrent CNVs in cancer have been found to amplify oncogenes such as ERBB2 and eliminate tumor suppressor genes such as PTEN [2]. The Cancer Genome Atlas (TCGA) [85] has compiled copy number data for hundreds of patients in a number of different cancers. The large number of samples from TCGA have provided the statistical power to identify recurrent aberrations from copy number data.

We now describe the array-based sequencing technology to detect CNVs. The result of this technology is that we get $\log_2$ ratios of the proportion of copies in the test genome compared to a known reference genome at particular locations in the reference genome. For example, if the $\log_2$ ratio falls below zero for a region, this suggests that there are fewer copies of the test genome than the reference genome at this region.

1.1.1 Array Comparative Genomic Hybridization (aCGH)

Array Comparative Genomic Hybridization (aCGH) [6,81,113,114] is a widely-used experimental technique for measuring CNVs on a genome-wide scale. As the name suggests, aCGH involves the comparison of two different cell populations, test DNA fragments and reference DNA fragments. These two populations are differentially labeled with two different fluorophores (put simply, they are "colored" different colors). They are then mixed in equal proportions and introduced to an array that contains DNA single-stranded sequences, called *probes*, attached to the surface. These probes, which typically range between 50-75bp [93], include short sequences that map uniquely to the reference genome.

The single-stranded, fluorescent-tagged DNA fragments hybridize to the probes on the array if the probes are the reverse-complement of the fragment sequence. Multiple probes with the same sequence are placed on the array in the same location, providing multiple fragments to hybridize near each other. Measurements of the test-to-reference fluorescence ratio at each location on the array identify locations in the test genome that are present in lower, higher, or similar copy in the reference genome. Typically, the $\log_2$ ratio of the colors is reported. Plotting these ratios along the originating locations of each probe on the reference genome produce a *copy number profile* of the test genome in terms of reference locations (Figure 1.2). From these copy number profiles, algorithms typically infer duplications and deletions (see Chapter 2 for a further discussion).

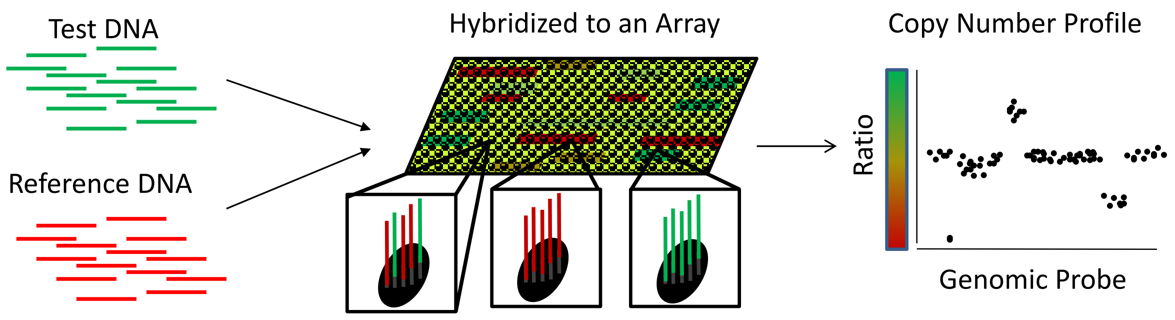

Figure 1.2: **Array comparative genomic hybridization (aCGH).** Fluorophore-labeled DNA from a test and a reference genome are hybridized to an array, and the fluorescence is interpreted as a ratio that represents the relative test:reference copy number.

Obstacles While aCGH has been used successfully to identify many CNVs in human genomes, there are some limitations to this technology. First, technologies to measure copy number (such aCGH) fail to detect other copy-neutral SVs, such as balanced inversions and translocations. Second, array-based methods are inherently limited by the size and origin of the probes used to hybridize the DNA. These probes must come from unique regions of the reference genome, and they may be many kilobases apart [113]. Since SVs are often found in repetitive regions [55] and the human genome itself is quite repetitive [127], array-based methods have difficulty designing probes that capture all of the structural variation in mammalian genomes.

1.2 SV Detection using DNA Sequencing Data

As the cost of whole-genome sequencing decreases, detecting SVs from DNA sequencing data has become an attractive option because sequencing technologies overcome the array-based limitations described above. However, the current cost of whole-genome DNA sequencing has limited the number of sequenced genomes. Efforts to accumulate large numbers of whole-genome sequenced samples such as the 1,000 Genomes Project [32] and TCGA [85] are currently in progress. Until there are enough fully-sequenced whole genomes, identifying recurrent SVs using sequencing data will have to wait. Instead, we focus our efforts on identifying SVs in single genomes from sequencing data. Since SVs are relatively rare (but functionally important) in the human genome, the general SV detection framework uses the concept of *whole genome resequencing* [12]. Resequencing takes sequenced fragments of an individual's genome (the *target genome*) and aligns them to another completely sequenced genome called a *reference genome*. From the alignments to the reference genome, we may identify differences between the target genome and the reference genome. Identifying these differences is complicated by a number of factors, including sequencing error in the target genome and alignment error when aligning sequences from the target genome to the reference genome. The presence of repetitive sequences in mammalian genomes provides an additional challenge when detecting human SVs [127].

Below we describe the different types of sequencing technologies, and then discuss the sequencing protocols that use linking, or pairing information between sequenced portions of DNA.

1.2.1 Next-Generation DNA Sequencing Technologies

Sequencing technologies have undergone many paradigm shifts in the past few decades, and each technology has substantial tradeoffs in terms of throughput, cost, read length and accuracy (Table 1.1). Early Sanger based methods [126] produce long, accurate reads, but have relatively low throughput. In the early 2000s, a new phase of sequencing technologies aimed to sequence a larger number fragments in a single experiment more quickly and cheaply than traditional Sanger sequencing. These second-generation, or *next-generation*, technologies are categorized as massively parallel sequencers that are able to sequence large amounts of DNA more cheaply than Sanger methods. Three of the most widely used next-generation platforms come

Name	Read Length (bp)	Per-Based Error Rates	Linked Reads?	Throughput (Per 24 hours)	Example Genome	References
First Generation						
Sanger Sequencing	600-1000	0.001%	✓	1.2Mb [107]*	J. Craig Venter	[73, 102, 131]
Second Generation						
Roche/454	330 avg (up to 700)	0.1-1.07%		1.3Gb	Namibian Male	[43, 79, 89, 129]
ABI/SOLiD	50	0.06%	✓	4.5Gb	Charcot-Marie-Tooth Patient	[82, 89]
Illumina HiSeq	100 (150 for Rapid Run)	1.4-2%	✓	54.5Gb	Yak Genome	[79, 91, 102, 117]
Third Generation (Single Molecule Sequencing)						
Helicos	24-70 (32 avg)	3.58%		4.6Gb	Stephen R. Quake	[89, 116]
Pacific Biosciences	1500 avg (up to 14Kb)	13-15%	✓	1.2Gb	Cholera	[29, 46, 118]
Complete Genomics	35 with 100Kb inserts	0.001%	✓	N/A**	Acral Melanoma Tumor	[70, 108, 141]
Ion Personal Genome Machine	200	1.71%	✓	1.9-2.4Gb (100bp Runs)	E. Coli	[79, 80, 87, 118]
Oxford Nanopore	>48Kb	4%		Unk.	Lambda Phage***	[98]*

*Reported in advertising articles and press releases.

**Complete Genomics uses an in-house instrument - the sample is sent to the company.

***The entire genome of the lambda phage was sequenced as one fragment.

Table 1.1: **Sequencing technologies.** Values are first taken from recent sequencing experiments, review articles, and benchmark comparison papers. "Linked Reads" denote paired-end, mate-pair, or strobe sequencing capabilities. Example genomes provide a sense of some of the genomes being sequenced, and are not necessarily considered the "state of the art" experiments with the technologies.

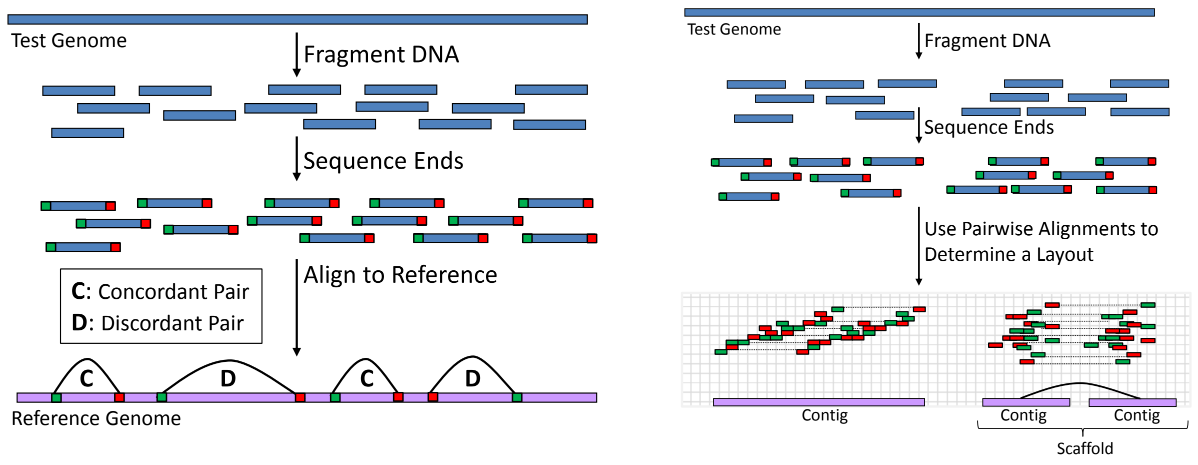

Figure 1.3: **Applications with Paired-End Data.** a test genome is sheared into DNA fragments, and the ends of these fragments are sequenced from the ends inwards denoting a left (green) and right(red) read for each fragment. (Left) In genome resequencing, the reads are mapped to a reference genome. The expected distance between the mapped pairs is determined empirically by the distribution of mapped distances. Pairs that have the expected mapped distance with reads in the proper orientation are concordant pairs, and pairs that are unexpected in terms of mapped distance, read orientation, or both are discordant pairs. (Right) In *de novo* assembly, an ordering of the reads (a *layout*) is determined from pairwise alignments of all reads. From the layout, we identify contiguous sequenced regions, called *contigs*, which may be linked to form *scaffolds* using the pairing information from individual reads in the contigs.

from Illumina, Roche/454, and Life Technologies. Though they have drastically different library preparation and sequencing chemistry, they all rely on massively parallel sequencing to produce orders of magnitude more bases per day than Sanger sequencing. Whole genomes have been sequenced with all of the next-generation technologies, and data cohorts such as the Cancer Genome Atlas [9, 85, 94] and 1000Genomes [32] collect samples sequenced by a variety of different technologies from different sequencing centers. Next-generation sequencers have been extensively discussed a number of reviews [83, 89, 102, 116, 131] and benchmarking papers [43, 70, 79, 80, 91, 118].

Measuring SVs using Paired-End Sequencing Data A sequencing protocol that is particularly useful for SV identification is *paired-end*, or mate-pair sequencing [13, 153], which is available from a number of different sequencing platforms (Table 1.1). In paired-end sequencing, DNA is sheared into longer fragments and the *ends* of each fragment are sequenced, producing a pair of *reads*. Resequencing approaches [12] align these reads to a reference genome, and the mapped distance between the aligned reads are observed (Figure 1.3 Left). The empirical distribution of the mapped distances is used to determine the “expected” fragment lengths. Most of the paired reads map to the reference genome with the expected distance and in the expected orientation, implying that there is no difference between the test genome and the reference genome at these mapped locations. We call these pairs *concordant* pairs. However, a small number of paired reads do not map to the reference genome with the expected distance and in the expected orientation.

These *discordant* pairs suggest an SV, where the test genome and the reference genome are different at these mapped locations.

Obstacles Identifying SVs from paired-end data is complicated by a few factors. Many SVs are associated with repeated sequences in the genome [62, 66], and the same read may align to many locations in the genome. Additionally, reads might not align correctly to the reference (due to errors, novel sequence in the test genome, or reads that span the rearranged breakpoints). To handle these obstacles, most methods to detect SVs cluster paired reads that indicate the same SV, and assign a higher confidence if more paired reads *support* the variant. However, these issues result in a large number of incorrect inferred SVs, despite reasonably good sensitivity when identifying known SVs [26, 52, 65, 133]. These methods are ultimately limited by the sequencing technology; in particular paired-end data does not completely disambiguate read alignments to the reference.

1.2.2 Third-Generation DNA Sequencing Technologies

Emerging single-molecule sequencing technologies, coined "third-generation" technologies, have kept the cost of sequencing low while being able to sequence longer fragments than next-generation technologies (Table 1.1. The exception is Helicos, which sequences 32bp on average; Helicos is sometimes referred to as a Next-Generation sequence technology, and was the first successful single-molecule sequencer available on the market. Pacific Biosciences has recently introduced Single Molecule Real Time (SMRT) sequencing, a DNA sequencing method that directly observes DNA polymerase as it synthesizes DNA [39, 67]. Currently, Pacific Biosciences has the ability to sequence fragments 1.5Kb on average, with some reads exceeding 14Kb (Michael Schatz, Biology of Genomes 2012). However, this long fragment length comes at the cost of significantly higher sequencing error rates of 13%. Additionally, the Pacific Biosciences sequencing error model is distinctly different from previous from other technologies because insertion and deletion errors are observed more frequently than nucleotide substitution errors. Other technologies like Oxford Nanopore promise the benefit of long reads with only a slight increase in the per-base error rates, but no official data has been published with this technology.

Measuring SVs using Strobe Sequencing Data Pacific Biosciences have developed a new protocol called *strobe sequencing* [142]. A strobe consists of multiple *subreads* from a single contiguous fragment of DNA (Figure 1.4). These subreads are separated by a number of "dark" nucleotides, called *advances*, whose identity is unknown. Conceptually, strobes generalize the concept of paired-end sequencing by allowing more than two subreads for each DNA fragment. Strobes are many kilobases on average, and strobes up to 20Kb with about 1Kb of total sequenced content have been obtained to-date (Ali Bashir, personal communication).

Despite the high sequencing error rates, strobe sequencing may help SV detection in two ways. First, additional subreads provide more unique locations to "anchor" strobes that have subreads in repetitive regions.

Second, a single strobe with more than two subreads may detect and characterize multiple SVs with complex, nested organizations, when paired-end data cannot distinguish nested variants. This is useful because variants have been observed to be found near each other in rearrangement hotspots [130].

Obstacles The error rates in the subreads mean that there may be multiple alignments to the reference genome, and the difficulty becomes choosing the correct alignment. Next-generation sequencing technologies also have dealt with the issue of multiple alignments because of the short read length coupled with the repetitive nature of the reference genome, but this obstacle is exaggerated with the higher error rates from Pacific Biosciences. Another issue when we have higher error rates is that it is more difficult to identify the precise SV coordinates because there sometimes are incorrect alignments that overlap the true ones.

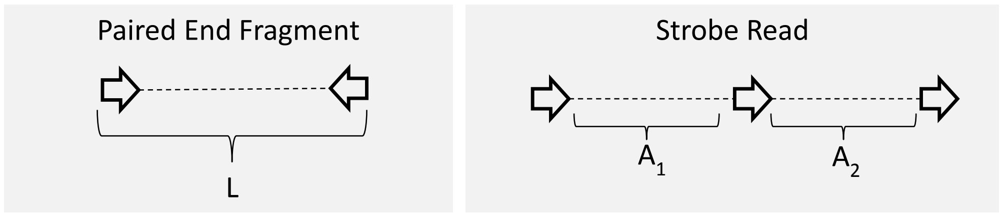

Figure 1.4: **Paired-End Sequencing vs. Strobe Sequencing.** In paired end sequencing, the ends of a DNA fragment are sequenced inwards from a fragment of length L. In strobe sequencing (as in this 3-strobe), the subreads are sequenced in the same orientation, creating three subreads and two advances (A_1 and A_2).

1.3 *De novo* Assembly of Individual Genomes

Sequencing cancer genomes invites the detection of a broader range of SVs compared to CNV-detection methods such as aCGH. However, the large number of SVs that appear in cancer genomes potentially limits the power of SV detection using a reference genome. Further, there might be inherent bias in the reference genome, and aligning to the reference is less accurate than, say, aligning to a normal (non-tumor) genome from the same patient. Thus, our last goal is to piece together highly-rearranged genomes without the use of a reference genome. The task of determining the underlying sequence represented by a set of sequenced reads is called *de novo* assembly, and it is computationally NP-hard [95]. As in the resequencing problem, mate-pair sequencing is often used for *de novo* assembly (Figure 1.3 Right). However, the reads are aligned to each other rather than to a reference genome, creating pairwise alignments of subreads. These pairwise alignments are then assembled to produce contiguous sequences called *contigs*. The pairing information from fragments where pairs are assembled into different contigs produce linked contigs which we call a *scaffold*.

Many assembly methods have been developed specifically for handling short read lengths in next-generation sequencing, and they have been extensively reviewed [90, 115, 127, 164]. Most algorithmic approaches to genome assembly construct a graph to represent overlaps between reads and use various heuristics to find a path through the graph that corresponds to the genome sequence; see Chapter 5 for a detailed description of these methods.

Obstacles While many *de novo* assemblers now exist, *de novo* assembly using next-generation sequencing still faces many challenges, including overcoming the repeats in the human genome using the short read lengths [3]. Additionally, sequencing errors in the reads (which vary by technology) pose an obstacle for *de novo* assembly. While whole-genome assemblies are now feasible, they are highly fragmented. Fragmented assemblies are particularly problematic for structural variation studies because many structural variants are found in regions that are difficult to assemble [62]. Three reasons for highly-fragmented assemblies are underrepresented (low-coverage) regions of the target genome, sequencing errors in the reads, and repetitive regions in the target genome.

Potential Benefits of Strobe Sequencing The advantages of single-molecule sequencing, including longer read lengths and multiple subreads (in the case of Pacific Biosciences), may help simplify and disambiguate the graph constructions in current *de novo* assemblers. However, these advantage are reduced by higher sequencing error rates. The anticipated sequencing error rates from single-molecule sequencers make many seed-based algorithms, such as de Bruijn graph assemblers, impractical. However, many of these methods are under-utilizing a key piece of information when constructing the graph - the pairing information. Most of the graph-based methods incorporate pairing information *after* the graphs are constructed; instead, we wish to simultaneously use the overlap information and the pairing information throughout the assembly method.

1.4 Summary of Contributions

In this section, we describe the research contributions of this dissertation. These points are expanded upon in the subsequent chapters.

CNV Detection in Multiple Cancer Genomes [125]. We present an algorithm called Neighborhood Breakpoint Correlation (NBC) to identify recurrent breakpoints in copy number data from technologies such as aCGH. This algorithm is comprised of the following novel concepts and components:

1. Some types of structural variation, such as fusion genes and gene truncations, require that the SV is conserved at the end points, or *breakpoints*, across multiple copy number profiles. Other methods typically identify recurrent intervals, rather than the precise recurrent breakpoints that our algorithm identifies.

2. Obtaining a single set of breakpoints for a copy number profile is not always useful for identifying recurrent breakpoints across multiple individuals, due to both the noise in the copy number profile

and the breakpoint variability across different individuals. Thus, we adapted a Bayesian changepoint algorithm [78] to compute the probability of a breakpoint given the entire copy number profile, rather than reporting an optimal set of breakpoints.

3. We developed a statistic that computes the statistical significance of a conserved breakpoint occurring in a subset of individuals given the breakpoint probabilities in a copy number profile. We generalize this statistic to compute the probability of a conserved breakpoint occurring within a specified window of adjacent probes (i.e. within a gene). Finally, we use this statistic to compute the probability of pairs of recurrent breakpoints that occur in a subset of individuals.

We apply NBC to two different datasets and obtain the following results:

1. We first apply NBC to a prostate cancer dataset, where we anticipate finding a single fusion gene, TMPRSS2-ERG, that is known to be present in prostate cancer. We consider the TMPRSS-ERG fusion gene a positive control.
   (a) Our statistic finds the TMPRSS-ERG fusion gene as the only statistically significant pair of genes that are not found near known structural variants, with a p-value of 2.7×10^{-10} after multiple hypothesis correction.
   (b) We show that using a single, optimal segmentation of the data does not produce breakpoints that fall within the TMPRSS2 and ERG gene regions for all patients with the TMPRSS2-ERG fusion. Further, we show that another recently-published Bayesian segmentation algorithm, BCP [41], provides much noisier breakpoint probabilities than our method for the copy number profiles containing the TMPRSS2-ERG fusion.
2. We apply NBC to hundreds of copy number profiles from glioblastoma (brain) tumors, and infer a number of candidate fusion genes and gene truncations.
   (a) We find that the phosphatase PTPN12 is an interacting gene partner with one third of all fusion gene candidates. We hypothesize that the disregulation of PTPN12 might be important for the progression of glioblastoma. While this hypothesis has yet to be experimentally validated, a recent study discovered that PTPN12 is a tumor suppressor gene in breast cancer [137].

SV Detection with Strobe Sequencing Data (A Parsimony Approach) [124]. We present a new method to detect SVs from strobe sequencing data. To our knowledge, this is the first method for SV detection from strobe sequencing data (or, more generally, sequencing data with more than two linked reads). Our algorithm is comprised of the following novel concepts and components:

1. We generalize the problem formulation in [52] to the t-strobe framework, where we have t linked DNA sequences, called subreads, from each strobe. This formulation aims to find the minimum number of SVs that describe the strobe alignments.

2. Due to the higher error rates of strobe sequencing, multiple alignments from a single subread must be considered. Other methods for paired-end data have made this observation [52, 119, 134], but using multiple alignments for paired-end data does not seem to significantly improve performance [134].

3. We construct a graph representation of the solution space, where a solution to the formulated problem is a subgraph subject to some constraints. This subgraph can be found by solving a flow problem on the graph, and we describe an Integer Linear Program (ILP) that returns an optimal solution. The flow problem is similar (but not identical) to the fixed-charge multi-commodity network flow problem [35].

We applied our method to simulated strobe sequencing data, since real strobe sequencing data was unavailable at the time. We generated simulated 3-strobe data and compared the strobes to paired-end datasets constructed from the 3-strobe subreads, which simply destroys the multiple links for strobes. From this simulation, we obtained the following results:

1. We find that the pairing information provided by the multiple links in strobes more than doubles the specificity in the prediction of deletions over the paired-end datasets.

2. Inversions are particularly difficult to identify due to highly-repetitive regions near the breakpoints. We find that strobes and paired-end datasets recover the same number of true inversions but strobes reduce the number of false inversions by 78%.

3. Strobes identify translocations where the breakpoints lie in repetitive regions, and this result is consistent when averaged over ten randomly-generated strobe datasets.

SV Detection with Strobe Sequencing Data (A Probabilistic Approach) [In preparation]. We found that the algorithm described above did not consider some useful pieces of information from the strobe data. Thus, we created a probabilistic framework for SV detection using strobes. This algorithm is comprised of the following novel concepts and components:

1. To describe the space of possible SVs, we create a graph that is based on the overlaps of the alignments that indicate an SV. The graph uses concepts similar to GASV [133], but GASV describes maximal sets of strobe alignments that are consistent with a SV breakpoint. Instead, our graph formulation concisely describes *all* possible sets of strobe alignments that are consistent with a single SV breakpoint.

2. We derive a probabilistic model for selecting at most one alignment for each strobe based on three observations:

    (a) Repetitive regions will have many incorrect strobe alignments that collectively support an SV. The parsimony assumption of minimizing the number of SVs falls apart when many repetitive regions are present in the genome, since these incorrect SVs will be selected. Since mammalian genomes are highly repetitive, this poses a problem for SV detection in humans.

(b) The alignment quality (the edit distance between a subread and the reference) is highly-variable due to the sequencing error rate. Additionally, with an error rate of 13-15$\%$ we *expect* some number of mismatches in the alignment (a 500bp subread alignment with zero errors, for example, is suspicious).

(c) A strobe might not have a correct alignment to the reference genome; this may arise due to many sequencing errors, novel sequence in the test genome, or a subread that spans an SV breakpoint.

3. Our probabilistic model allows strobes with different numbers of subreads (i.e mixing 3-strobes and 4-strobes) as well as mixing strobe and paired-end data with different sequencing error rates. The model also allows split-read alignments (alignments where the first part aligns to one portion of the reference and the second part aligns to another portion).

We applied the probabilistic method to a variety of simulated and real data from human genomes and obtain the following results:

1. The probabilistic method applied to low-coverage simulated strobe sequencing data (15$\%$ error rate) from two simulations (one on Chr17, one on Chr1) outperforms the previous approach in terms of both sensitivity and specificity.

2. Low-coverage (5X) strobe sequencing data performs comparably to high-coverage (30X) paired read data. Additionally, mixing high-coverage pairs with low-coverage strobes improves performance over high-coverage pairs alone (improving sensitivity) and low-coverage pairs alone (improving specificity).

3. We compared our results to high-coverage (30X) paired read data run with other SV prediction methods [119, 133, 134]. We find that the probabilistic method, when applied to paired read data, enriches for correct alignments better than the other methods.

4. We obtained real strobe sequencing data from Pacific Biosciences of four 40Kb regions of the human genome that reportedly contain SVs. We recover the single "true positive" in each case with the probabilistic method, despite the fact that the two inversions have breakpoints that lie in nearly-identical regions.

***De novo* Assembly with Strobe Sequencing.** We have developed a small-scale *de novo* assembly algorithm for strobe sequencing data. This algorithm is ideal for small-scale assembly and it can be used as a "finishing" step when most of the layout is known from other methods. It is comprised of the following novel concepts and components:

1. We define an assembly "score" that uses pairwise alignment qualities, and we formulate the assembly problem as a maximization problem for assemblies that are consistent with the linking information provided by strobes.

2. The constraints on the strobes may be written as a set of linear inequalities, and we design an Integer Linear Program (ILP) to find an optimal solution. The number of variables scales quadratically with the number of reads, so we have developed an iterative algorithm that greedily samples strobes and uses the ILP to assemble them at each step.

3. A partial assembly consists of a series of scaffolds, or assembled sequenced linked by pairing information. We represent these scaffolds as a special type of strobe, called a *derived strobe*, with different subread and advance lengths than the other strobes.

We applied our method to simulated strobe sequencing data from four Bacterial Artificial Chromosomes (BACs) that have been hard to assemble with paired-end sequencing. We obtain the following results:

1. The ILP-based assemblies on sequenced data with various sequencing error rates produce a single scaffold for all datasets. Further, there are at most 7 contigs in any assembly, and 100% of the strobes are placed in the final layout in three of the four datasets.

2. We compared our method to a de-Bruijn assembler called Velvet [159, 160], which performs poorly when assembling datasets with large error rates. However, this is expected because de-Bruijn graphs rely on a seed size for determining overlaps.

3. We compared our method to an overlap-graph assembler Bambus [140], which performs much better than Velvet. However, it still produces much more fragmented assemblies than the ILP assembler and assembles a smaller percentage of the reads, indicating that the assembly is less complete than the ILP assembler.

Chapter 2

CNV Detection with Applications to Cancer

Array comparative genomic hybridization (aCGH) [6, 81, 113, 114] is a common technique for measuring CNVs (Figure 1.2). An aCGH experiment produces a *copy number profile* of test:reference copy number ratios. If we have copy number profiles from multiple individuals with the same type of cancer, the goal is to identify SVs that appear in multiple copy number profiles; these *recurrent* SVs may be important in the progression of the cancer of interest.

Previous methods have primarily focused on on identifying recurrent segments (adjacent probes with similar copy number); however there are a number of drawbacks to this approach. In this chapter, we introduce a novel algorithm called Neighborhood Breakpoint Correlation (NBC) to identify recurrent breakpoints in copy number data. Here, a *breakpoint* is a pair of adjacent probes where the copy number ratio changes from low to high or high to low. The work in this chapter is taken from [125], and was originally presented at the Second Annual RECOMB Satellite Workshop on Computational Cancer Biology (RECOMB-CCB) in 2010.

The power of NBC to detect recurrent breakpoints, particularly breakpoints that appear in a small percent of cancer patients, relies on a large number of copy number profiles to provide statistical significance. The Cancer Genome Atlas (TCGA) [85] is an effort to understand the genomic changes in cancer by analyzing hundreds of tumors from a variety of cancer patients using many available technologies. Initiated in 2005 with a pilot project of two cancers, seventeen different cancers are now being analyzed with up to 588 publicly-available samples for each cancer type. The ultimate goal is to acquire hundreds of samples from over twenty different cancers.

As a controlled experiment, we first apply NBC to aCGH data from 36 primary prostate tumors and infer 12 CNVs, including one gene truncation and one fusion gene which is the well-known TMPRSS2-ERG fusion gene. Next, we apply NBC to 227 glioblastoma (GBM) tumors from TCGA and infer 91 CNVs, including 23 gene truncations and 33 fusion genes. Additionally, we predict 35 germline CNVs from 107 available matched blood samples from GBM patients. A number of the inferred somatic CNVs in GBM involve the protein phosphatase PTPN12, suggesting that deregulation of PTPN12 via a variety of rearrangements is common in glioblastoma. We note that NBC is readily adapted to analyze copy number profiles obtained from next-generation DNA sequencing data [28, 157].

2.1 Related Work

Copy number profiles contain experimental artifacts, including measurement error in the probes. Additionally, isolated probes might be duplicated or deleted; we are not interested in these single-probe aberrations because we wish to detect larger SVs. Thus, a first step in CNV detection often includes smoothing, or segmenting the copy number profile into a set of segments of equal copy number. Numerous segmentation methods exist to convert copy number profiles into *segmentations*; some methods ([56, 77, 105, 144]) return a single optimal segmentation, while others ([41], [47], [112]) produce a posterior probability of a change in copy number at each location over all possible segmentations of the copy number profile. See [69] for a comparative analysis of segmentation algorithms.

To identify recurrent CNVs in cancer, current methods [10, 15, 37, 150, 163] develop statistics to find recurrent aberrant intervals from a set of copy number profiles. They first identify aberrant intervals within each sample and then combine the aberrant intervals to produce statistically significant recurrent aberrant intervals. These methods are good for identifying some types of recurrent CNVs, such as deletions and amplifications that harbor tumor suppressor genes or oncogenes.

To identify recurrent CNVs in cancer, statistical methods are used to combine copy number profiles from a set of individuals with the same cancer type. Several methods have been introduced to identify recurrent CNVs. CoCoA [10] segments and scores candidate aberrations for each individual, and then combines the scores using a binomial order statistic to find significant aberrations. GISTIC [15], STAC [37], and DiNAMIC [150] all determine significant aberrations from segmented copy number profiles using various permutation tests. These methods all suffer from the obstacle that a single segmentation is used for each copy number profile, when it is often the case that many different segmentations of the copy number profile are equally plausible. However, if all the highly-probable segmentations are similar, then this should not affect the significance of large recurrent aberrations. CMDS [163] is a method that aims to identify recurrent aberrations directly from copy number profiles by finding correlation blocks in a correlation matrix.

However, other CNVs such as unbalanced translocations are not characterized by a single interval. Additionally, the targeted gene in a recurrent SV might not be within the aberrant interval, but rather at the ends of the interval. *Fusion genes*, for example, are a type of CNV where a rearrangement fuses portions of two genes to produce a functioning gene [92]. One documented fusion gene, the TMPRSS2-ERG fusion, arises from a 3Mb deletion on Chr21 in patients with prostate cancer [92, 139]. Recurrent CNVs such as fusion genes and unbalanced translocations are characterized by the end points, or *breakpoints* of the recurrent rearrangement. Identifying recurrent breakpoints is much more sensitive to the resulting segmentation of a copy number profile because we wish to find common loci of copy number change. Methods that use a single segmentation to describe a copy number profile might miss recurrent breakpoints that appear in other slightly-less-optimal segmentations. Additionally, there might be biological variability in the breakpoints, where there are multiple loci within a gene that may be used as a breakpoint that results in the same function.

2.2 The Neighborhood Breakpoint Correlation (NBC) Algorithm

The Neighborhood Breakpoint Correlation (NBC) algorithm takes, as input, aCGH data from many individuals and identifies recurrent breakpoints and pairs of recurrent breakpoints in a subset of the individuals (Figure 2.1). The first step in NBC, as in most aCGH analysis, is to segment each copy number profile into intervals of equal copy number. NBC uses a dynamic programming approach [78] to compute the probability $P(\mathbf{X}|\mathcal{A})$ of a copy number profile $\mathbf{X}$ given a segmentation $\mathcal{A}$. NBC then employs a stochastic backtrace to compute the posterior probability $P(\mathcal{A}|\mathbf{X})$. Using this approach one can derive the segmentation $\hat{\mathcal{A}}$ with maximum probability, but more importantly, one can compute the posterior probability of events of interest over *all possible segmentations* of the data. In particular, we compute the probability of a breakpoint between each pair of adjacent probes, as well as the probability of a breakpoint within a fixed interval or probes (e.g. from a gene region).

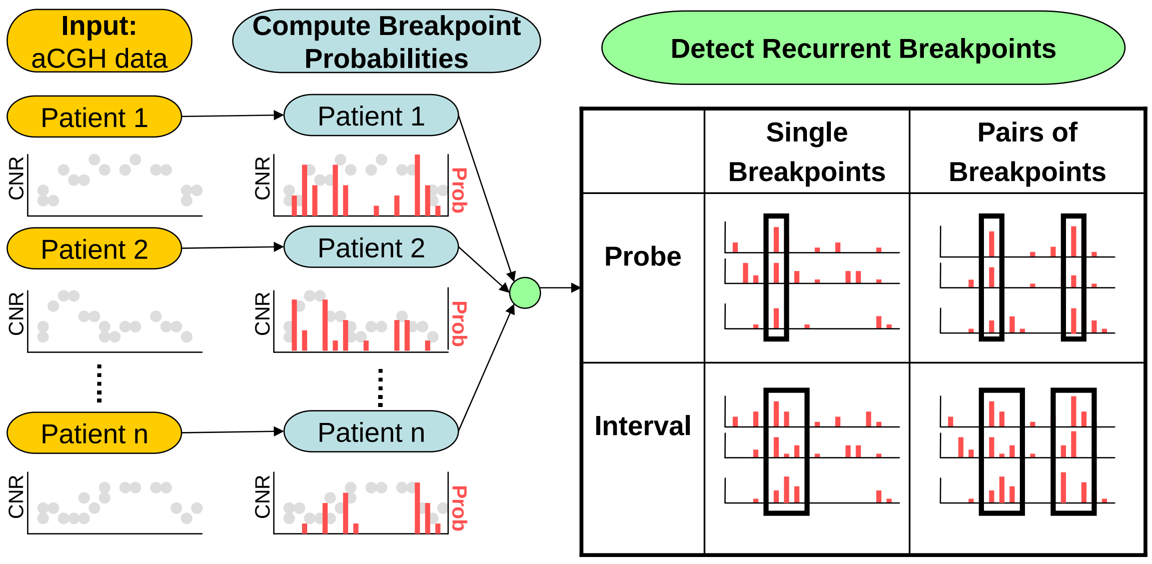

Figure 2.1: **The Neighborhood Breakpoint Correlation (NBC) algorithm.** NBC consists of two steps: computing breakpoint probabilities and recurrent breakpoint detection. Copy number ratios (CNRs) derived from aCGH data from multiple individuals are segmented using a Bayesian change-point algorithm that computes the probability of a breakpoint between adjacent probes (in red). The breakpoint probabilities are then combined to detect recurrent breakpoints (black rectangles). We identify recurrent breakpoints that occur between adjacent probes as well as recurrent breakpoints that occur within a set of probes defined by a genomic interval. To detect CNVs, we identify pairs of recurrent breakpoints.

The second step of NBC is to combine breakpoint probabilities in each individual to determine breakpoints that appear in multiple individuals. Similar to [10], we use a binomial order statistic [36] to compute a p-value for the event that k or more individuals share a breakpoint between two adjacent probes. We then extend this breakpoint score to consider pairs of breakpoints that are shared by multiple individuals. Finally, we also define a score for a breakpoint that may occur anywhere within an interval of adjacent probes (e.g. a gene) that is shared by multiple individuals. We detail each of these two steps in the following sections.

2.2.1 A Probability Model for Segmentation and Breakpoint Analysis

A probabilistic formulation of the segmentation problem assigns a probability to each possible segmentation of $\mathbf{X}$, and the probability of other events, such as a breakpoint occurring at a particular locus, are readily computed from this model. Probabilistic segmentation approaches have been previously applied to CNV detection [7, 41, 47, 162], but we found that these methods either: require a finite number of copy number levels (as in the Bayesian Hidden Markov method of [47]); focus on probabilistic model selection rather than an explicit probabilistic model for the segmentation itself [162]; or do not perform well on high-resolution oligonucleotide arrays (see §2.3.1 for a comparison to [41] and [105]).

Our algorithm is based on the change-point model described in [78]. Consider a copy number profile $\mathbf{X} = (X_1, \ldots, X_n)$, where X_i is the $\log_2$ ratio of test:reference DNA at the ith probe. We assume that the test genome consists of an unknown number of segments K with corresponding segment values $\Theta = \{\theta_1, \ldots, \theta_K\}$. Following the usual assumptions for aCGH data [7, 41, 47, 112, 162], we assume that each X_i is normally distributed with mean μ_i and variance σ^2, where the variance σ^2 is a hyperparameter that must be estimated from the data and will be described later. The mean μ_i equals θ_s if probe i lies within to segment s. Further, we assume that X_i from different segments are independent. Let l_j denote the number of probes in segment j, and let $k_{\max}$ denote the maximum number of segments in the test genome.

We define the *breakpoint sequence* $\mathbf{A} = (A_1, \ldots, A_{K+1})$, where A_v is the index of the probe at the start of the $v + 1st$ segment and $A_{K+1} = n$ is a “dummy” breakpoint signifying the end of the sequence (i.e. there are K+1 breakpoints representing K segments in $\mathbf{A}$). Thus,

$$A_v = \sum_{j=1}^{v} l_j + 1 \ \text{ for } 1 \leq v \leq K. \tag{2.1}$$

The unknown variables in our model are the breakpoint sequence $\mathbf{A}$, the number of segments K, and the segment values Θ.

We assume a priori that Θ is independent of $\mathbf{A}$ and K, and we assume that the segment values $\theta_s \in \Theta$ are independent. We select a conjugate prior $\theta_s \sim \mathcal{N}(\mu_0, \sigma_0^2)$ on the segment values, where μ_0 represents equal copy number between the test and the reference genomes. We assign a prior on breakpoints sequences $\mathbf{A}$ such that all $\mathbf{A}$ with K segments are equally likely,

$$P(\mathbf{A}|K) = \binom{n}{K}^{-1}. \tag{2.2}$$

Additionally, we assign a prior on the number of segments K such that there is a probability of $1/2$ of a single segment and the remaining values of K are equally likely,

$$P(K) = \begin{cases} 1/2 & K = 1 \\ 1/(2(k_{\text{max}})) & 1 < K \leq k_{\text{max}} \end{cases} . \tag{2.3}$$

Note that these priors do not make any strong assumptions on the data, except that we expect a single segment with probability $1/2$. For notational convenience, let $X_{[i:j]} = (X_i, \ldots, X_j)$, $X_{(i:j]} = (X_{i+1}, \ldots, X_j)$, and $X_{[i:j)} = (X_i, \ldots, X_{j-1})$. From the priors $P(\mathbf{A}|K)$ and $P(K = k)$ and the hyperparameter values, the probability of $\mathbf{X}$ is

$$\begin{aligned} P(\mathbf{X}) &= \sum_{k=1}^{k_{\text{max}}} P(\mathbf{X}|K{=}k)P(K{=}k) & (2.4) \\ &= \sum_{k=1}^{k_{\text{max}}} P(K{=}k)\left[\sum_{\mathcal{A}:||\mathcal{A}||=k} \int P(\mathbf{X}, \mathbf{A}{=}\mathcal{A}|\Theta, K{=}k)P(\Theta)d\Theta\right] & (2.5) \\ &= \sum_{k=1}^{k_{\text{max}}} P(K{=}k)\left[\sum_{\mathcal{A}:||\mathcal{A}||=k} \prod_{v=1}^{k} \int P(X_{[A_v:A_{v+1})}|\theta_v)P(\theta_v)d\theta_v\right] . & (2.6) \end{aligned}$$

Here $||\mathcal{A}||$ is the length (number of breakpoints) of $\mathcal{A}$, $P(\mathbf{X}|K = k)$ is the probability of the data $\mathbf{X}$ given that the test genome is divided into k segments, and $P(X_{[A_v:A_{v+1})}|\theta_v)$ is the probability that $X_{[A_{v-1}:A_v)}$ consists of a single segment. The product in Equation (2.6) results from the segment independence assumption.

Analytical Computation of $P(\mathbf{X})$

The choice of a conjugate prior for $P(\theta)$ allows the integral

$$\int P(X_{[A_v:A_{v+1})}|\theta_v)P(\theta_v)d\theta_v = P(X_{[A_{v-1}:A_v)}|\mu_0, \sigma_0^2, \sigma^2, K = 1) \tag{2.7}$$

to be analytically computed.

For simplicity, we show the derivation for computing the integral for the entire dataset $\mathbf{X}$, where $|\mathbf{X}| = n$; however in practice this can be done on any sequence of points $X_{[i,j)}$. Following Theorem 9.6 in [148],

$$P(\mu|\mathbf{X}, \mu_0, \sigma_0^2, \sigma^2, K=1) = N(\mu^*, (\sigma^*)^2), \text{ where} \tag{2.8}$$

$$\mu^* = \frac{n\overline{\mathbf{X}}\sigma_0^2 + \mu_0\sigma^2}{n\sigma_0^2 + \sigma^2}, \tag{2.9}$$

$$\sigma^* = \sqrt{\frac{\sigma_0^2\sigma^2}{n\sigma_0^2 + \sigma^2}}, \text{ and} \tag{2.10}$$

$$\overline{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} X_i. \tag{2.11}$$

Each observation X_i is normally distributed with mean μ_i and the prior $P(\mu)$ is also normally distributed. We can multiply the $n+1$ normals and produce

$$P(\mathbf{X}|\mu, \mu_0, \sigma_0^2, \sigma^2, K=1)P(\mu) = \frac{1}{(2\pi)^{(n+1)/2}\sigma^n\sigma_0} \times \exp\left[\frac{-1}{2\sigma^2}\sum_{i=1}^{n}(X_i - \overline{\mathbf{X}})^2\right] \tag{2.12}$$

$$\times \exp\left[\frac{-1}{2}\left(\frac{n(\overline{\mathbf{X}} - \mu)^2}{\sigma^2} + \frac{(\mu - \mu_0)^2}{\sigma_0^2}\right)\right]. \tag{2.13}$$

Using the identity

$$\sum_{i=1}^{n}(X_i - \mu)^2 = \sum_{i=1}^{n}(X_i - \overline{\mathbf{X}})^2 + n(\overline{\mathbf{X}} - \mu)^2, \tag{2.14}$$

expanding the exponent in (2.13) results in

$$\frac{-1}{2}\left[\left(\frac{n(\overline{\mathbf{X}}-\mu)^2}{\sigma^2}+\frac{(\mu-\mu_0)^2}{\sigma_0^2}\right)\right] \tag{2.15}$$

$$=\frac{-1}{2}\left[\mu^2\left(\frac{n\sigma_0^2+\sigma^2}{2\sigma^2\sigma_0^2}\right)-2\mu\left(\frac{n\sigma_0^2\overline{\mathbf{X}}+\mu_0\sigma^2}{\sigma^2\sigma_0^2}\right)+\left(\frac{n\sigma_0^2\overline{\mathbf{X}}^2+\mu_0^2\sigma^2}{\sigma^2\sigma_0^2}\right)\right] \tag{2.16}$$

$$=\frac{-1}{2}\left[\mu^2\left(\frac{1}{(\sigma^*)^2}\right)-2\mu\left(\frac{n\sigma_0^2\overline{\mathbf{X}}+\mu_0\sigma^2}{\sigma^2\sigma_0^2}\right)+\left(\frac{n\sigma_0^2\overline{\mathbf{X}}^2+\mu_0^2\sigma^2}{\sigma^2\sigma_0^2}\right)\right] \tag{2.17}$$

$$=\frac{-1}{2(\sigma^*)^2}\left[\mu^2-2\mu\left(\frac{n\sigma_0^2\overline{\mathbf{X}}+\mu_0\sigma^2}{n\sigma_0^2+\sigma^2}\right)+\left(\frac{n\sigma_0^2\overline{\mathbf{X}}^2+\mu_0^2\sigma^2}{n\sigma_0^2+\sigma^2}\right)\right] \tag{2.18}$$

$$=\frac{-1}{2(\sigma^*)^2}\left[\mu^2-2\mu\mu^*+(\mu^*)^2-\left(\frac{n\sigma_0^2\sigma_0^2(\mu_0-\overline{\mathbf{X}})^2}{(n\sigma_0^2+\sigma^2)^2}\right)\right] \tag{2.19}$$

$$=\frac{-1}{2(\sigma^*)^2}\left[\mu^2-2\mu\mu^*+(\mu^*)^2-\left(\frac{(\sigma^*)^2 n(\mu_0-\overline{\mathbf{X}})^2}{n\sigma_0^2+\sigma^2}\right)\right] \tag{2.20}$$

$$=\frac{-1}{2(\sigma^*)^2}\left[\mu^2-2\mu\mu^*+(\mu^*)^2\right]+\left(\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right). \tag{2.21}$$

$$\tag{2.22}$$

Plugging this into $P(\mathbf{X}|\mu,\mu_0,\sigma_0^2,\sigma^2,K=1)P(\mu)$ yields

$$P(\mathbf{X}|\mu,\mu_0,\sigma_0^2,\sigma^2,K=1)P(\mu)=\frac{\exp\left[-\frac{\sum_{i=1}^n(X_i-\overline{\mathbf{X}})^2}{2\sigma^2}+\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right]}{(2\pi)^{(n+1)/2}\sigma^n\sigma_0}\times\exp\left[\frac{-(\mu-\mu^*)^2}{2(\sigma^*)^2}\right]. \tag{2.23}$$

Marginalizing over μ finally gives the probability of the data $\mathbf{X}$ given that it is a single segment; we need to multiply the top and bottom by $\sqrt{2\pi(\sigma^*)^2}$ to accomplish this.

$$P(\mathbf{X}|\mu_0,\sigma_0^2,\sigma^2,K=1) = \int P(\mathbf{X}|\mu,\mu_0,\sigma_0^2,\sigma^2,K=1)P(\mu)d\mu \tag{2.24}$$

$$= \frac{\sqrt{2\pi(\sigma^*)^2}\exp\left[-\frac{\sum_{i=1}^n (X_i-\overline{\mathbf{X}})^2}{2\sigma^2}+\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right]}{(2\pi)^{(n+1)/2}\sigma^n\sigma_0} \tag{2.25}$$

$$\times \int \frac{1}{\sqrt{2\pi(\sigma^*)^2}}\exp\left[\frac{-(\mu-\mu^*)^2}{2(\sigma^*)^2}\right] \tag{2.26}$$

$$= \frac{\sqrt{2\pi(\sigma^*)^2}\exp\left[-\frac{\sum_{i=1}^n (X_i-\overline{\mathbf{X}})^2}{2\sigma^2}+\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right]}{(2\pi)^{(n+1)/2}\sigma^n\sigma_0} \tag{2.27}$$

$$= \sqrt{\frac{\sigma^2\sigma_0^2}{(n\sigma_0^2+\sigma^2)}}\frac{\exp\left[-\frac{\sum_{i=1}^n (X_i-\overline{\mathbf{X}})^2}{2\sigma^2}+\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right]}{(2\pi\sigma^2)^{n/2}\sigma_0} \tag{2.28}$$

$$= \sqrt{\frac{\sigma^2}{(n\sigma_0^2+\sigma^2)}}\frac{\exp\left[-\frac{\sum_{i=1}^n (X_i-\overline{\mathbf{X}})^2}{2\sigma^2}+\frac{n(\mu_0-\overline{\mathbf{X}})^2}{2(n\sigma_0^2+\sigma^2)}\right]}{(2\pi\sigma^2)^{n/2}} \tag{2.29}$$

Define the noise-to-signal ratio $w(l_v) = \sigma^2/(n_v\sigma_0^2+\sigma^2)$, which depends on the length l_v of the segment. We can now write down the probability that $\mathbf{X}$ is generated from $k > 1$ segments:

$$P(\mathbf{X}) = \sum_{k=1}^{k_{\max}} P(\mathbf{X}|K=k)P(K=k) \tag{2.30}$$

$$= \sum_{k=1}^{k_{\max}} P(K=k)\left[\sum_{\mathcal{A}:||\mathcal{A}||=k}\prod_{v=1}^{k}\int P(X_{[A_v:A_{v+1})}|\theta_v)P(\theta_v)d\theta_v\right] \tag{2.31}$$

$$\propto \sum_{\mathbf{A}:||\mathbf{A}||=k}\left[\prod_{v=1}^{k}\left[\sqrt{w(l_v)}\exp\left[\frac{\sum_{i=A_{k-1}}^{A_k-1}(X_i-\overline{X}_{[A_{k-1}:A_k)})^2+w(l_v)l_v(\mu_0-\overline{X}_{[A_{k-1}:A_k)})^2}{2\sigma^2}\right]\right]\right]. \tag{2.32}$$

However, calculating $P(\mathbf{X}|K=k)$ in this way requires summing over all possible breakpoint sequences $\mathbf{A}$ and is computationally infeasible. A dynamic program allows the efficient computation of this term.

Dynamic Program

Let $P(X_{[i:j]}|k)$ be the probability of observing $X_{[i:j]}$ given that it is generated from k different segments. We compute this $P(X_{[1:j]}|k)$ for $1 \leq k \leq k_{\text{max}}$ and $1 \leq j \leq n$ as follows:

$$P(X_{[1:j]}|k) = \begin{cases} \sum_{v<j} \left[P(X_{[1:v)}|k-1)P(X_{[v:j]}|1)\right] & 1 < k \leq j \\ 0 & k > j. \end{cases} \tag{2.33}$$

$$\text{Base case: } P(X_{[i:j]}|1) = \int P(X_{[i:j]}|k=1, \theta, \sigma^2)P(\theta)d\theta \qquad 1 \leq i \leq j \leq n. \tag{2.34}$$

The final row of the dynamic programming table contains $P(\mathbf{X}|K=k)$ for $1 \leq k \leq k_{\text{max}}$, which is used in Equation (2.4) to compute $P(\mathbf{X})$.

Recursive Sampling

We describe a sampling strategy that will generalize to the computation of the probability of breakpoints that lie within an interval or pairs or breakpoints in §2.2.2. We use $P(\mathbf{X}|K=k)$ as well as the base case $P(X_{[i:j]}|1)$ and intermediate terms $P(X_{[1:j]}|k)$ in the dynamic program to sample exact and independent breakpoint sequences $\mathbf{A}$ using a backward sampling technique [78]:

1. Draw $K=k$ from $P(K=k|\mathbf{X})$, determined by inverting $P(\mathbf{X}|K=k)$ using Bayes Rule.
2. Set $A_{k+1} = n$.
3. Draw $A_k, A_{k-1}, \ldots, A_1$ recursively using the conditional distributions computed by the recurrences in (2.33). Given A_q, the location of the beginning of the qth segment, the distribution of A_{q-1} is obtained as follows:

$$P(A_{q-1} = j|\mathbf{X}, A_q = m) = \frac{P(X_{[1:j]}|q-1)P(X_{(j:m]}|1)}{P(X_{[1:m]}|q)}. \tag{2.35}$$

From a set of breakpoint sequences sampled in proportion to $P(\mathbf{A}|\mathbf{X})$, we determine the probability of a breakpoint occurring between two adjacent probes by counting the proportion of samples that contain a breakpoint at that locus. Other probabilities derived from these sampled breakpoint sequences are described in subsequent sections.

Runtime analysis. The base cases $P(X_{[i:j]}|1)$ require $O(n^2)$ computations and the dynamic program requires $O(nk_{\text{max}})$ computations; thus computing $P(\mathbf{X}|K=k)$ is achieved in $O(n(n+k_{\text{max}}))$ time. All computations necessary to sample a breakpoint sequence $\mathbf{A}$ are already computed in the dynamic program, so sampling is linear in the number of breakpoints K drawn from $P(\mathbf{X}|K=k)$.

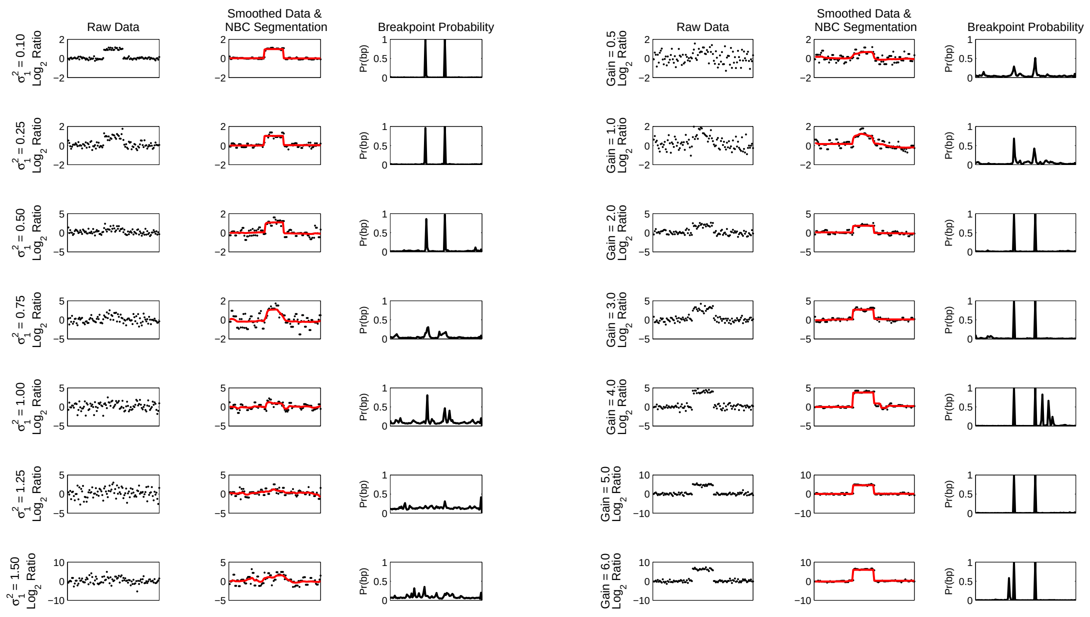

Figure 2.2: **Hyperparameter estimation on simulated datasets.** The first column shows the raw data for a particular aberration $\log_2$ ratio. The second column shows the smoothed data and the average NBC segmentation (red). The third column shows the breakpoint probability at each location. (Left) A single artificial chromosome from Simulation #1 with gaussian noise $N(0, \sigma_1^2)$ for $\sigma_1^2 = 0.1, 0.25, 0.5, 1, 1.25$ or 1.5. (Right) A single artificial chromosome from Simulation #2 with gaussian noise $N(0, 0.5)$ for aberration $\log_2$ ratios of $0.5, 1, 2, 3, 4, 5$ and 6.

Hyperparameter Estimation

The segmentation algorithm relies on setting values for the hyperparameters μ_0 (the baseline mean), σ_0^2 (the segment variance), and σ^2 (the variance from experimental error). We describe how to estimate these from aCGH $\log_2$ ratios. First, we set μ_0 to be the median value of the $\log_2$ ratios. To estimate the variances σ_0^2 and σ^2, we break the dataset into sliding windows of 10 probes. Let V be the median of the sample variances of the windows, and let M be the maximum absolute distance between the sample means of the windows and μ_0. We set the variance from experimental error $\sigma^2 = 2V$ and the segment variance $\sigma_0^2 = M^2$.

The datasets we segment have a wide range of variances in the $\log_2$ ratios (the Glioblastoma experiments, for example, have $\log_2$ ratio variances that range from 0.0084 to 7.5591). To test the sensitivity of our choice of hyperparameters, particularly σ^2 and σ_0^2, we performed two simulations similar to the simulations of [41].

Simulation #1 We generated an artificial chromosome of length 100 with a 40 probe single-copy gain ($\log_2$ ratio of 1) placed in the center. We then introduced various amounts of gaussian noise $N(0, \sigma_1^2)$ in the probe measurements, where $\sigma_1^2 = 0.1, 0.25, 0.5, 1, 1.25,$ or 1.5. For each value of σ_1^2, we generated 100 such chromosomes.

Simulation #2 We generated an artificial chromosome of length 100 with gaussian noise $N(0, 0.5)$ in the probe measurements. We then introduced a 40 probe aberration at various $\log_2$ ratios: $0.5, 1, 2, 3, 4, 5$, and 6. For each $\log_2$ ratio, we generated 100 such chromosomes.

A representative sample of the datasets for Simulation #1 and Simulation #2 are shown in Figure 2.2.

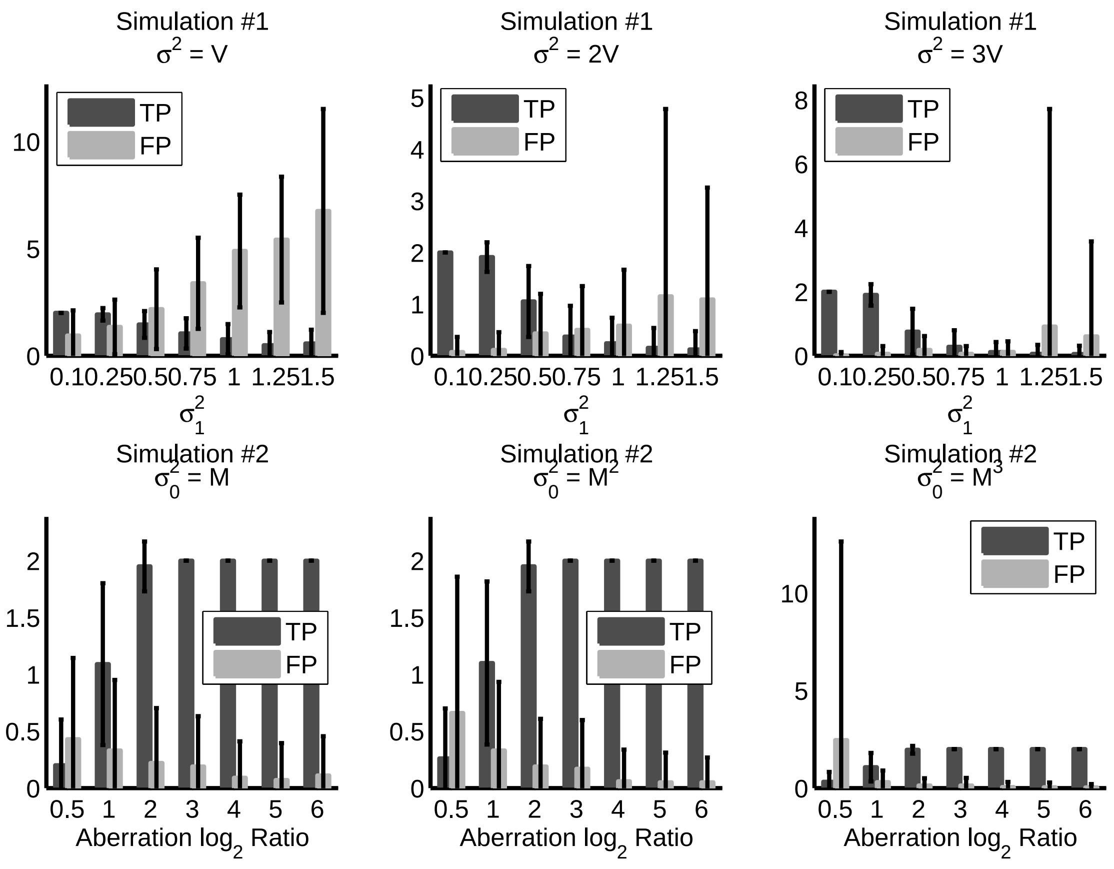

Figure 2.3: **Hyperparameter Sensitivity Analysis.** The number of true positive (TP) breakpoints (0,1, or 2) and the number of false positive (FP) breakpoints for Simulation #1 and Simulation #2 over various values of variance parameters σ_0^2 (top row) and σ^2 (bottom row). The bars are averaged over 100 iterations for each simulation, and error bars indicate 1 standard deviation.

We ran NBC on datasets from the two simulations with different estimates for the variances σ_0^2 and σ^2, which will be described in the following paragraphs. To assess the quality of the resulting set of inferred breakpoints, we consider probe locations with Pr(breakpoint) ≥ 0.5 to be an inferred breakpoint. We assume that an inferred breakpoint detects a true breakpoint if the inferred breakpoint location is ≤ 2 probes away from the true breakpoint location. We count the number of true positive breakpoints (0, 1, or 2). Additionally, we count the number of false positive breakpoints for each dataset. We average the true positives and false positives over the 100 artificial chromosomes.

Simulation #1 has a fixed aberration $\log_2$ ratio, so we set the segment variance $\sigma_0^2 = M^2$ and we test three different values of σ^2: V, $2V$, and $3V$ (Figure 2.3 top row). Compared to our estimated value of $\sigma^2 = 2V$, the number of true positives is similar to $\sigma^2 = V$ and $\sigma^2 = 3V$ at low noise σ_1^2. As σ_1^2 increases, setting $\sigma^2 = V$ results in more false positives compared to our estimate $\sigma^2 = 2V$, while setting $\sigma^2 = 3V$ results in fewer total inferred breakpoints, including true positives. Thus, we estimate σ^2 as $2V$.

Simulation #2 has fixed noise, so we set the variance of the experimental error $\sigma^2 = 2V$ and test three different values of σ_0^2 : M,M^2, and M^3 (Figure 2.3 bottom row). With the exception of a large standard deviation for $\sigma_0^2 = M^3$, the number of true positives and false positives are similar for the three different estimates of σ_0^2.

The simulations with large noise σ_1^2 reasonably demonstrate that when the segment variance is smaller than the noise variance, the measurement noise obfuscates any subtle changepoints in the data. Thus, when $\sigma \leq 3\sigma_0$, we do not segment the data and immediately report 0 breakpoints.

2.2.2 Identifying Recurrent Breakpoints

After sampling breakpoint sequences for a set of individuals, we identify recurrent breakpoints that appear in many individuals at the same genomic locus. Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be a set of copy number profiles from m individuals, where $S_j = (X_1, \ldots, X_n)$ is the copy number profile for individual j. We assume that the same array probes are used for each individual, i.e. the ith probe in individual S_j is at the same location as the ith probe in individual $S_{j'}$. We analyze recurrent breakpoints at two levels of resolution:

1. *Recurrent probe breakpoints* occur between the same two array probes in a subset of individuals.

2. *Recurrent interval breakpoints* occur within the same interval of the genome in a subset of individuals.

In addition to analyzing these types of recurrent breakpoints, we also consider pairs of recurrent breakpoints to identify recurrent CNVs. Note that these pairs may indicate *intrachromosomal* CNVs, as in the case of classic copy number aberrations like duplications and deletions, or *interchromosomal* CNVs, as in the case of (unbalanced) translocations.

Recurrent Probe Breakpoints

For each probe, we define a score that measures the presence of a breakpoint in a subset of individuals. We design this score to account for the observation that the number of breakpoints in copy-number profiles, particularly in a set of cancer samples, is highly variable. That is, in a set of cancer samples, even from the same cancer type, there will typically be highly rearranged cancer genomes with many breakpoints, and less rearranged genomes with relatively few breakpoints. This variability in the number of breakpoints is maintained following our Bayesian segmentation approach – despite the fact that we use the same flat prior for each individual – because there is strong evidence to support a larger number of breakpoints in some samples. Since there is a greater chance of recurrent breakpoints occurring randomly in a collection of highly rearranged genomes than a collection of less rearranged genomes, it is advantageous to consider the number of breakpoints in each profile when scoring recurrent breakpoints. Because the variability of number of breakpoints across different individuals is typically not well matched by a standard distribution, one approach is to use a permutation test that preserves the number and probability of breakpoints in each profile while permuting their location. We instead derive a score for recurrent probe breakpoints based on a binomial order statistic [10, 36]. This score first normalizes the breakpoint probability at each probe in each individual according to the breakpoint probabilities across all probes in individual. These normalized values are then combined across multiple individuals to produce a recurrent breakpoint score.

Let b_i be the event that a breakpoint lies between probes i and $i+1$; $P(b_i|S_j)$ is the *breakpoint probability* at probe i in individual S_j, and is computed by counting the proportion of sampled breakpoint sequences $\mathbf{A}$ that have a breakpoint between i and $i+1$. Let $\rho_j(i)$ be the fraction of probes with a higher breakpoint probability than probe i in individual S_j (the normalized rank of probe i):

$$\rho_j(i) = \frac{|\{g : P(b_g|S_j) \geq P(b_i|S_j)\}|}{n}. \tag{2.36}$$

Let π be a permutation of the individuals $\mathcal{S}$ such that $\rho_{\pi_1}(i) \leq \rho_{\pi_2}(i) \leq \ldots \leq \rho_{\pi_m}(i)$. For $1 \leq h \leq m$, we wish to determine the probability that h or more individuals have a breakpoint at location i. Because of our normalization of the breakpoint probabilities in each sample, under the null hypothesis the individual scores $\rho_j(i)$ are independent and uniformly distributed in $[0, 1]$. Thus, the probability that h or more individuals have a breakpoint at location i is given by the tail of the binomial distribution with success probability $\rho_{\pi_h}(i)$. Thus, the p-value for the probe location i is

$$p(i) = \min_{h=h_{\min},\ldots,m} \left[\sum_{j=h}^{m} \binom{m}{j} \rho_{\pi_h}(i)^j (1 - \rho_{\pi_h}(i))^{m-j}) \right], \tag{2.37}$$

where we are only interested in scoring those breakpoints that are present in at least $h_{\min}$ patients. Note that because the binomial order statistic is computed from the empirical distribution ρ_j of breakpoint probabilities in each sample, the relative magnitude of the breakpoint probability is not used in the computation. Despite this loss of information, we found that the binomial order statistic produced reasonable results on real data (See Results below) and was more efficient than a permutation test.

Finally, we assume that a recurrent breakpoint is also conserved in the direction of the copy number change: all samples with a recurrent breakpoint are either breakpoints that go from relatively low copy number to high copy number of vice versa. A breakpoint sequence $\mathbf{A}$ defined a segmentation, and we use the mean values of each segment to determine the direction of copy number change. The copy number change is positive if the mean of the segment to the right of the breakpoint is higher than the mean of the segment to the left. We test both cases for each recurrent breakpoint, doubling the number of hypotheses we test. We control the False Discovery Rate (FDR) using the method of Benjamini and Hochberg [11].

Recurrent Interval/Gene Breakpoints

We extend our approach to find recurrent breakpoints that lie within a genomic interval W. This interval can represent the set of probes within a gene region, for example. Unlike the recurrent probe breakpoint calculation above, where each probe was a priori equally likely to contain a breakpoint, intervals that contain more probes are a priori more likely to contain a breakpoint than intervals that contain fewer probes. To account for this, we use a log-odds score that is defined as follows. Let $b \in W$ be the event that one or more breakpoints lie between any pair of adjacent probes within W. Similarly, let $b \notin W$ be the event that no breakpoint lies between any adjacent probes within W. The log-odds score $\ell_j(W)$ that patient S_j contains a breakpoint within W is

$$\ell_j(W) = \log \frac{P(S_j|b \in W)}{P(S_j|b \notin W)} = \log \frac{P(b \in W|S_j)}{P(b \notin W|S_j)} \frac{P(b \notin W)}{P(b \in W)}. \tag{2.38}$$

The conditional probabilities $P(b \in W|S_j)$ and $P(b \notin W|S_j)$ describe probabilities over all possible segmentations of the copy number profile S_j. $P(b \in W|S_j)$ is determined by sampling breakpoint sequences $\mathbf{A}$ as described in §2.2.1 and counting the number of samples that contain one or more breakpoints in the interval W. $P(b \notin W|S_j)$ is then simply $1 - P(b \in W|S_j)$. The scaling factor $\frac{P(b \notin W)}{P(b \in W)}$ is computed by counting the number of ways to place breakpoints such that none of them lie in W:

$$\begin{aligned} P(b \notin W) &= \sum_{k=1}^{k_{\max}} P(K = k) P(b \notin W | K = k) && (2.39)\\ &= \sum_{k=1}^{k_{\max}} P(K = k) \left(\frac{\binom{n-|W|}{k}}{\binom{n}{k}} \right), && (2.40)\\ P(b \in W) &= 1 - P(b \notin W). && (2.41) \end{aligned}$$

Here, the last term in (2.40) counts the number of ways to choose k breakpoints that do not lie in W. As in the recurrent breakpoint computation above, we use the binomial order statistic to combine log-odds scores across patients. First, in an analogous computation to (2.36) we normalize the log-odds scores using the empirical cumulative distribution, which produces the normalized rank of $\ell_j(W)$ for all j:

$$\rho_j(W) = \frac{|\{g : \ell_g(W) \geq \ell_j(W)\}|}{|\mathbf{W}|}. \tag{2.42}$$

Finally, using the $\rho_j(W)$ scores for each patient S_j we compute the p-value $\rho(W)$ using the binomial order statistic as in (2.37).

For the experiments below, we define the the copy number change for an interval W to be positive if at least 90% of the breakpoints within the interval are positive and negative if at least 90% of the breakpoints within the interval are negative. Otherwise, we do not call a breakpoint in W.

Pairs of Recurrent Interval/Gene Breakpoints

We identify pairs of non-overlapping recurrent interval breakpoints using a log-odds score similar to Equation (2.38) that scores two breakpoints occurring in intervals W_1 and W_2. An important case we will consider is when W_1 and W_2 are genes. Let $b \in W_1$ be the event that a breakpoint lies between any pair of adjacent probes within W_1, and Let $b' \in W_2$ be the event that a breakpoint lies between any pair of adjacent probes within W_2. We define the score for intervals W_1 and W_2 for a particular patient S_j:

$$\ell_j(W_1, W_2) = \log \frac{P(S_j | b \in W_1 \cap b' \in W_2)}{P(S_j | b \notin W_1 \cup b' \notin W_2)} \tag{2.43}$$

$$= \log \frac{P(b \in W_1 \cap b' \in W_2 | S_j)}{P(b \notin W_1 \cup b' \notin W_2 | S_j)} \frac{P(b \notin W_1 \cup b' \notin W_2)}{P(b \in W_1 \cap b' \in W_2)}. \tag{2.44}$$

Each term is computed similarly to (2.38). If W_1 and W_2 are on different chromosomes, the events $P(b \in W_1)$ and $P(b' \in W_2)$ are independent and Equations (2.40) and (2.41) are used to compute the scaling factor $\frac{P(b \notin W_1 \cup b' \notin W_2)}{P(b \in W_1 \cap b' \in W_2)}$. If the intervals are on the same chromosome then the events are dependent, and the quantities in $\frac{P(b \notin W_1 \cup b' \notin W_2)}{P(b \in W_1 \cap b' \in W_2)}$ are

$$P(b \notin W_1 \cup b' \notin W_2) = \sum_{k=1}^{k_{\max}} P(K = k) P(b \notin W_1 \cup b' \notin W_2 | K = k) \tag{2.45}$$

$$= \sum_{k=1}^{k_{\max}} P(K = k) \left(\frac{\binom{n-|W_1|}{k} + \binom{n-|W_2|}{k} - \binom{n-|W_1|-|W_2|}{k}}{\binom{n}{k}} \right), \tag{2.46}$$

$$P(b \in W_1 \cap b' \in W_2) = 1 - P(b \notin W_1 \cup b' \notin W_2). \tag{2.47}$$

The p-value $\rho(W_1, W_2)$ is computed by normalizing as in Equation (2.42) according to the empirical distribution of log-odds scores over all pairs of non-overlapping intervals and then using the binomial order statistic to determine the final p-value. Here, we test four hypotheses for each pair W_1 and W_2 by considering the four combinations of direction of copy number change: $\{(+,+), (-,-), (-,+), (-,-)\}$. Note that restricting W_1 and W_2 to each contain a single probe identifies pairs of recurrent probe breakpoints.

2.2.3 Predicting Structural Variants, Gene Truncations, and Fusion Genes

Our statistics for single recurrent breakpoints ($\rho(i)$ and $\rho(W)$) and pairs of recurrent breakpoints ($\rho(i,j)$ and $\rho(W_1, W_2)$) provide a flexible framework to infer particular rearrangement configurations. We classify inferred breakpoints into three groups: (1) structural variants, (2) gene truncations, and (3) fusion genes.

Structural Variants

Pairs of recurrent probe breakpoints may indicate germline or somatic rearrangements that have recurrent breakpoints at the highest resolution allowed by the spacing of probes. To identify these rearrangements, we compute the pairs of recurrent probe breakpoint statistic for every pair of probes within each chromosomal arm. Note that this limits the inferred structural variants to intrachromosomal rearrangements only.

Gene truncations

Recurrent breakpoints found within a single gene may indicate a gene truncation, resulting in the loss of functionality for a particular gene. To infer gene truncations, we compute the recurrent interval breakpoint detection statistic, using the set of gene regions from RefSeq as our intervals of interest.

Fusion Genes

Pairs of recurrent interval breakpoints found within genes suggest potential fusion genes. We compute pairs of recurrent interval breakpoints using all pairs of gene regions from RefSeq as our intervals of interest. Note that not all pairs of recurrent genes suggest functional fusion genes. For example, a rearrangement that joins the 3' end of one gene to the 3' end of another gene is typically not a functional fusion gene. Thus, we restrict our attention to pairs of interval breakpoints with particular configurations (Figure 2.4).

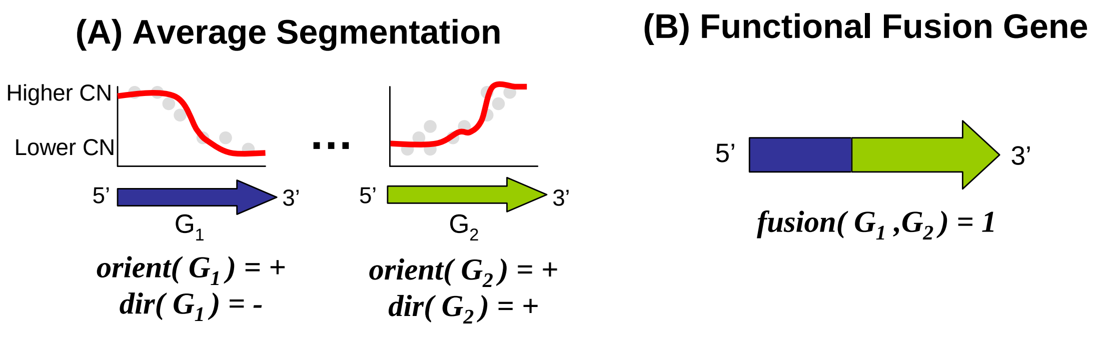

Figure 2.4: **Fusion Gene Configurations.** Fusion genes are pairs of recurrent genes that have the following configuration: (A) Each gene G_1 and G_2 has an associated orientation, $orient(G_1)$ and $orient(G_2)$. Additionally, each recurrent breakpoint has an associated change in relative copy number, $dir(G_1)$ and $dir(G_2)$. (B) A fusion gene joins the ends of G_1 and G_2 such that the 5' end of one gene is joined to the 3' end of the other gene.

Specifically, consider a pair of recurrent intervals G_1 and G_2 that represent gene regions. Each gene has an orientation, $orient(G_1) \in \{+,-\}$ and $orient(G_2) \in \{+,-\}$. Additionally, the breakpoint that lies within each recurrent interval has an associated direction of copy number change, $dir(G_1) \in \{+,-\}$ and $dir(G_2) \in \{+,-\}$. We assume that a fusion gene contains the 5' end of one gene joined to the 3' end of the other gene and thus satisfies the following rule:

$$fusion(G_1, G_2) = \begin{cases} 1 & \text{if } orient(G_1) \times dir(G_1) \neq orient(G_2) \times dir(G_2) \\ 0 & \text{otherwise.} \end{cases} \tag{2.48}$$

2.2.4 Filtering and Ranking Inferred SVs

We apply a number of additional steps to remove and prioritize inferred breakpoints for real data.

Removing Single Probe Aberrations

Single probe aberrations are segments consisting of a single probe. Since these are difficult to distinguish from experimental artifacts, we remove them from further consideration. Single probe aberrations are characterized by two large changes in copy number in adjacent probes, where the segments adjacent to this aberration have a similar copy number. We identify these probes and remove them from the analysis.

Removing Known CNVs

We remove inferred breakpoints that are near known CNVs. We say that a single probe is "near" a known CNV in the Database of Genomic Variants (DGV) [58] if it is within 10kb of a recorded copy number variant endpoint, and a gene region is "near" a known copy number variant if it is within 10kb of a recorded copy number variant endpoint. Additionally, a pair of intrachromosomal recurrent breakpoints are near a variant if at least one of the breakpoints is within 10kb of a recorded copy number variant endpoint and the mutual overlap between the prediction interval (defined by the pair of breakpoints) and the variant interval is greater than 50%.

Ranking Inferred Pairs of Breakpoints

Since fusion genes (and other recurrent pairs of breakpoints) are physically joined in the test genome, we expect the copy number of either side of the breakpoint to be the same. Thus, we rank inferred fusion genes by calculating the root mean squared difference (RMS) between the copy number levels of probes surrounding the breakpoint. For fusion genes, we know the configuration of the gene partners but we do not know exactly where the breakpoint lies. Thus, we determine the copy number on each side of the fusion as the average of the three flanking probes of the left gene partner and the three flanking probes of the right gene partner. If h patients have the breakpoint, determined by the argmax of Equation (2.37), $c_l^{(i)}$ is the left-flanking copy number of the fusion and $c_r^{(i)}$ is the right-flanking copy number of the fusion, then the RMS difference of the pair of conserved breakpoints is

$$RMS = \sqrt{\frac{1}{h}\sum_{i=1}^{n}(c_l^{(i)} - c_r^{(i)})^2}. \tag{2.49}$$

2.3 Results

We applied NBC to two aCGH datasets: a collection of 36 primary prostate tumors, and 227 glioblastoma (GBM) tumors. For each dataset, we computed recurrent probe breakpoints, recurrent gene breakpoints, pairs of recurrent probe breakpoints, and pairs of recurrent gene breakpoints.

2.3.1 Prostate Dataset

Breakpoint Type	**Rearrangement Type(s)**	# Predicted	# in DGV	# Novel
Recurrent Probes	Highly Conserved Breakpoints	80	66	14
Recurrent Genes	Gene Truncations	6	5	1
Pairs of Recurrent Probes	Germline or Somatic Structural Variants	38	28	10
Pairs of Recurrent Genes With Fusion Gene Config.*	Intrachromosomal Fusion Genes	2	1	1
	Interchromosomal Fusion Genes	2	2	0

*Novel pairs of recurrent gene breakpoints consistent with the fusion gene configuration (see § 2.2.3).

Table 2.1: **Predicted Recurrent Breakpoints in 36 Prostate Samples.** Breakpoint types are described in §2.2.3 and are explained by the indicated rearrangement type. '# Predicted' is the number of inferred breakpoints that are significant with FDR < 0.01. '# in DGV' counts the breakpoints near known structural variants in the Database of Genomic Variants (DGV). '# Novel' is the number of inferred breakpoints that are not near any known variant in DGV.

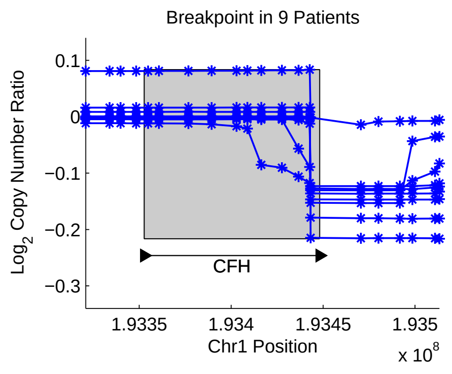

Figure 2.5: **A Predicted Gene Truncation in Prostate Cancer.** The Complement Factor H (CFH) gene on Chromosome 1 contains a recurrent gene breakpoint, suggesting the truncation of the 3' region in 9 individuals.

We applied NBC to Agilent aCGH data from a collection of 36 primary prostate tumors. Each sample contained copy number ratios for 235,719 aCGH probes that were mapped to the hg17 human reference genome. We examined recurrent gene breakpoints using the gene regions from 16,162 hg17 RefSeq genes. Table 2.1 reports the number of inferred variants, and tables listing the breakpoint coordinates and additional information are in Supplementary Tables 1 through 4 in [124]. We visualize inferred breakpoints by plotting the *average* segmentation for each of the individuals that were involved of the final p-value computations for recurrent breakpoints in Equation (2.37). The average segmentation is created by averaging the segment values Θ at each probe for the sampled breakpoint sequences $\mathbf{A}$.

We infer one novel gene truncation, which occurs in the Complement factor H (CFH) gene (Figure 2.5).

CFH encodes a protein that is secreted into the bloodstream and is essential for complement system regulation, and CFH polymorphisms are associated with macular degeneration [31]. From pairs of recurrent probes, we infer 10 novel variants, the most significant of which lies in the β-Defensin locus ((p-value=1.3×10^{-33}, Figure 2.6). β-Defensin genes have been associated with the risk of prostate cancer [57], and this locus lies near many known CNVs, complicating the study of nearby genes.

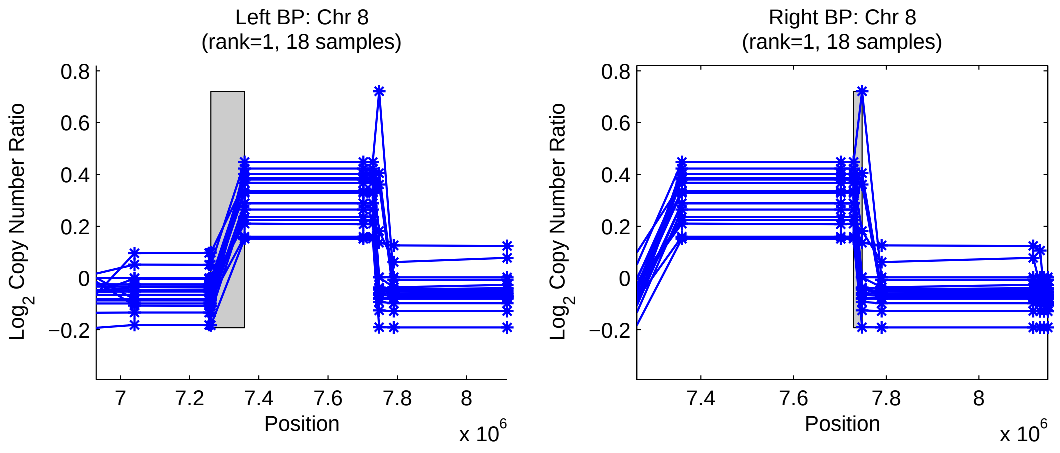

Figure 2.6: **A Predicted Rearrangement Highly Conserved at the Probe Level in Prostate Cancer.** This amplified region on Chromosome 8 lies in the β-Defensin locus, and the recurrent breakpoints are conserved at the probe level in 17 individuals. Arrows on the right indicating β-Defensin genes are approximate.

We infer only one fusion gene, the well-known TMPRSS2-ERG fusion gene, which we detect in 5 patients with a p-value of 2.7×10^{-10} (Figure 2.7). The TMPRSS2-ERG fusion gene has an RMS difference of 0.0759.

Comparison to Segmentation Approaches

The method by Erdman and Emerson, BCP [41], applies the change-point algorithm by Barry and Hartigan [7] to aCGH data using a different generative model than the segmentation portion of NBC. Their segmentations tend to have many more probes with high breakpoint probabilities than expected, even on relatively smooth data. Figure 2.8 shows the average segmentations and breakpoint probabilities averaged over the 5 patients that contain the TMPRSS-ERG fusion gene in Prostate cancer. While the fusion gene is clearly visible from the segmentations from BCP, the fusion gene cannot be clearly identified from the breakpoint probabilities of BCP.

To demonstrate the importance of breakpoint uncertainty in computing recurrent breakpoints, we compared inferred fusion gene to those obtained using a single segmentation for each individual. We segmented copy number profiles from each individual using Circular Binary Segmentation (CBS) [105] (Figure 2.9).

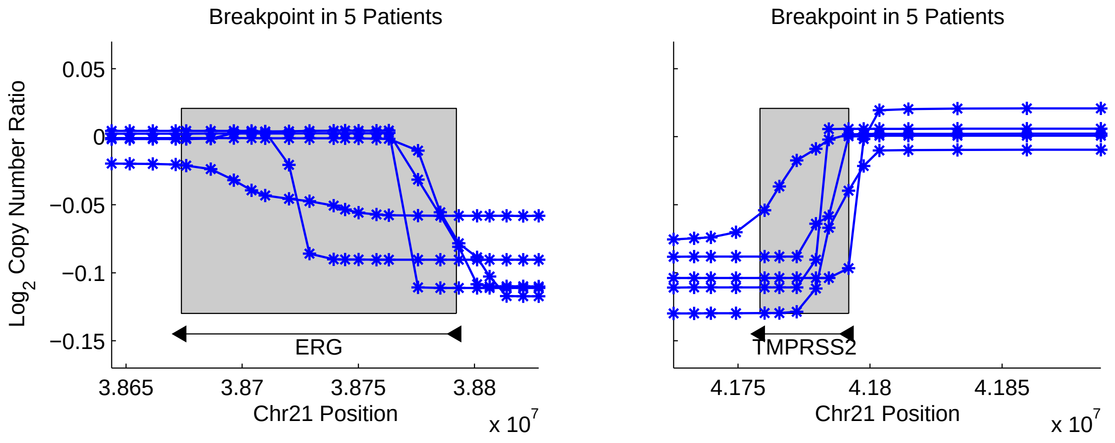

Figure 2.7: **The TMPRSS2-ERG Fusion Gene in Prostate Cancer.** We identify the TMPRSS2-ERG fusion gene in 5 prostate cancer patients. The mean segmentations for each patient (shown in blue) are computed by finding the segment parameters θ for each segmentation $\mathbf{A}$ drawn from the posterior distribution $P(\mathbf{A}|\mathbf{X})$ and then averaging these values across all segmentations.

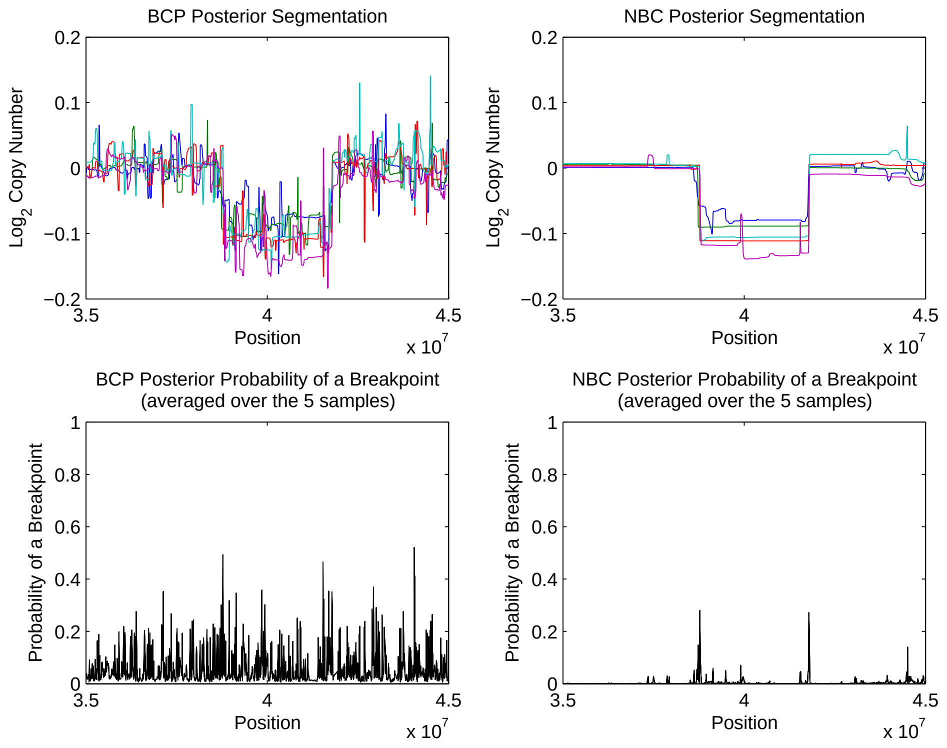

Figure 2.8: **A Comparison of Bayesian Segmentation Algorithms for aCGH data.** Bayesian segmentations for BCP (Left) compared to NBC (Right) for the patients that have the TMPRSS-ERG fusion gene. The mean segmentation is depicted on the top row, and the probability of a breakpoint $P(b_i)$, averaged over the 5 patients, is depicted on the bottom row.

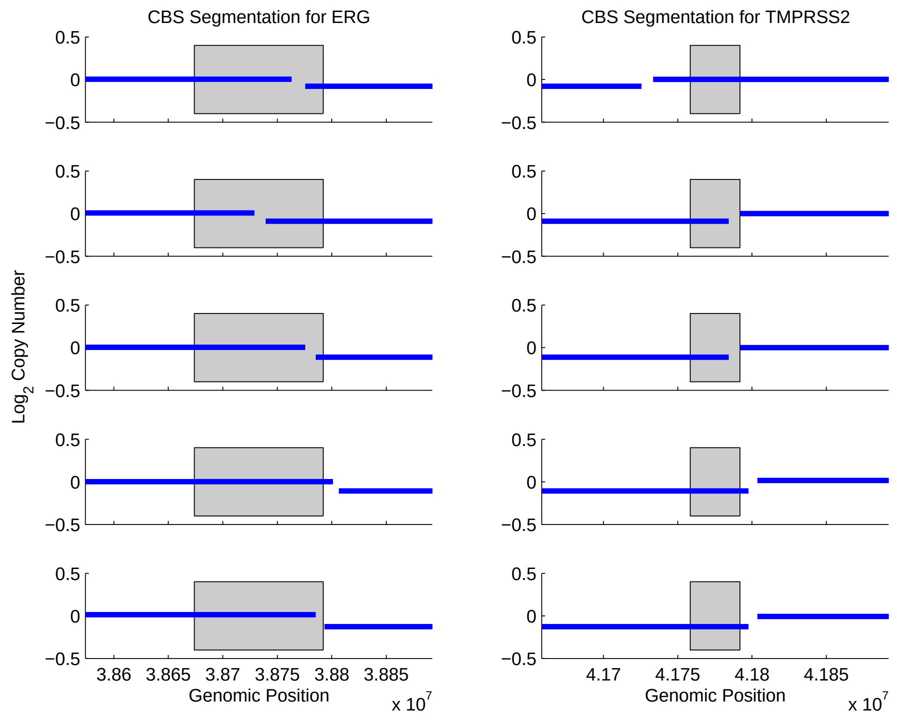

Figure 2.9: **Comparison with CBS segmentations**. We show the CBS segmentation for the 5 patients that have the TMPRSS-ERG fusion gene according to NBC. Only two of the five individuals have co-occurring breakpoints within the gene regions (gray boxes).

The CBS implementation in the R package DNAcopy was used to segment the prostate samples. The parameters used were the same as The Cancer Genome Atlas protocol [85]: the data was smoothed using a standard deviation smoothing technique (smooth.CNA with a smoothing region of 10), 10,000 hybrid permutations were used (nperm-10000,p.method="hybrid"), splits were undone using a standard deviation of 1 (undo.splits="sdundo"" undo.SD=1), and alpha was 0.01. Figure 2.9 shows the CBS breakpoint locations for the five individuals that NBC reports as having co-occurring breakpoints

CBS returns a single segmentation (and thus a set of breakpoints) for each individual. From these sets of breakpoints, for each pair of genes from the same chromosome, we counted the number of patients with a breakpoint in each gene. Only two individuals had a pair of breakpoints within TMPRSS2 and ERG from the CBS segmentations. Further, there are 5 inferred fusion genes that occur in two individuals after applying the filters described in §2.2.4, and zero predictions that occur in more than two individuals. Since no other common fusion genes in prostate cancer are known, we assume that these remaining inferred fusion genes are false positives. Thus, NBC is more sensitive and specific in fusion gene identification.

Rearrangement Type(s)	# Predicted	# in DGV	# in Blood	# Novel
Highly Conserved Breakpoints	538	343	13	189
Gene Truncations	92	69	23	23
Germline Structural Variants	88	53	N/A	35
Intrachromosomal Fusion Genes	75	45	5	7
Interchromosomal Fusion Genes	396	316	53	26

*FDR is increased to < 0.1 for blood samples.

**Novel pairs of recurrent gene breakpoints consistent with the fusion gene configuration (see § 2.2.3).

Table 2.2: **Predicted Recurrent Breakpoints in 227 GBM Samples and 107 Blood Samples.** Columns are described in Table 1, except for '# in Blood' which indicates the number of inferred breakpoints that also appear in the blood samples and are thus not considered somatic rearrangements.

2.3.2 Glioblastoma Dataset

We next applied our method to Agilent 244K aCGH data of glioblastoma (GBM) tumors from TCGA [85]. Data was collected from 233 GBM patients, including 227 tumor samples and 107 matched blood samples. Each sample contains 227,612 aCGH probes across the hg18 human reference genome. Gene regions from 16,162 hg18 RefSeq genes were used to determine recurrent gene breakpoints. Classification and filtering of breakpoints in the tumor samples were performed as above. Additionally, to restrict attention to somatic breakpoints we remove from consideration any recurrent breakpoints found in the tumor samples that also appear in the blood samples. When identifying recurrent probe breakpoints in the blood samples, we increase the False Discovery Rate (FDR) from 0.01 to 0.1 to more aggressively filter recurrent breakpoints in tumor samples. Table 2.2 reports the number of inferred variants, and tables listing the breakpoint coordinates and additional information are in Supplementary Tables 5 through 8 in [124].

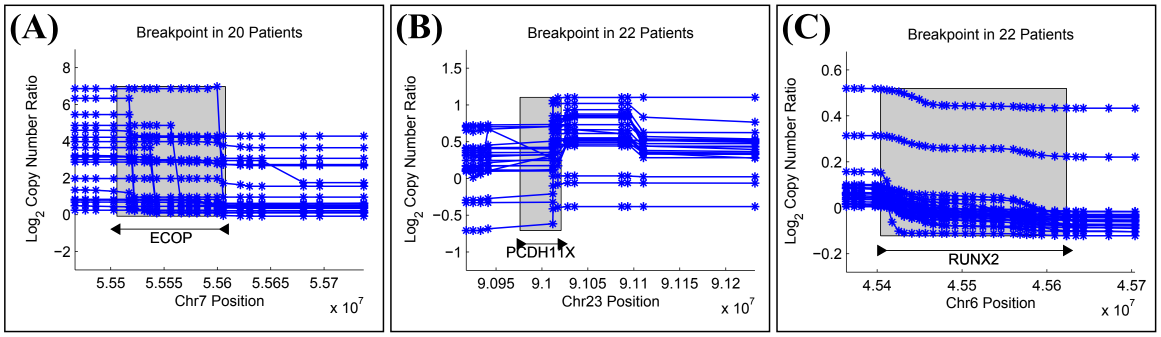

Figure 2.10: **Predicted Gene Truncations in GBM.** These three recurrent gene breakpoints found on Chromosome 7, Chromosome X, and Chromosome 6 respectively suggest truncations of genes associated with glioblastoma or other neuronal diseases. (A) The recurrent breakpoint in ECOP has a large change in copy number; this gene is near EGFR and is the breakpoint location for the EGFR amplification. (B) PCDHone1X appears to arise from a short deletion within a relatively amplified region, though the deletion breakpoint varies within the PCDHone1X gene region. (C) RUNX2 contains two probe locations with recurrent probe breakpoints that each have small copy number change at approximately 45.42Mb and 45.58Mb.

We infer 23 gene truncations from the tumor samples, three of which are shown in Figure 2.10. Each of these has some support in the literature for an association with glioblastoma or other neuronal diseases. ECOP is co-amplified with EGFR in glioblastoma as well as other cancers [5, 40], RUNX2 is expressed in glioblastoma cells [145], and PCDH11X is associated with late-onset Alzheimer's disease [19]. We also infer 33 fusion genes from the tumor samples. One of the inferred gene fusion involving INTS2 and MED13 might arise due to a tandem duplication whose breakpoints are within the two genes (Figure 2.11a). Another prediction involves PPP1R9A, which is an imprinted gene that appears in neuronal tissues and has been shown to be expressed in other embryonic tissues (Figure 2.11b). [97].

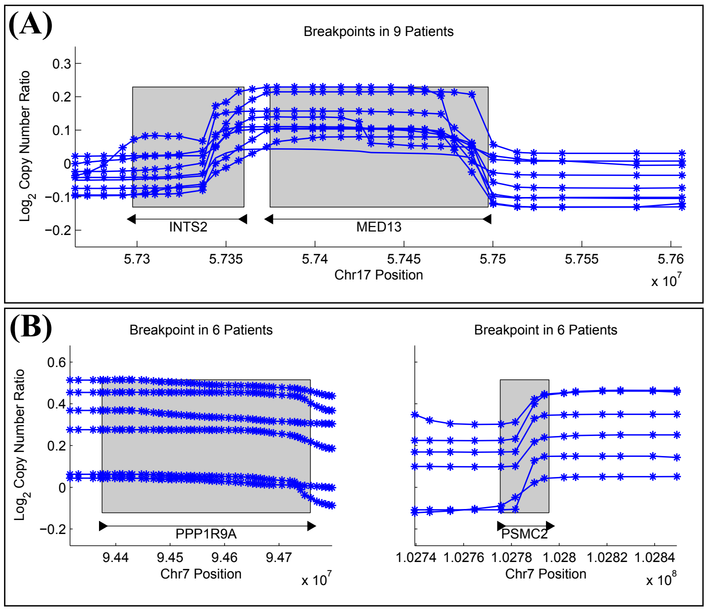

Figure 2.11: **Predicted Intrachromosomal Fusion Genes in GBM.** (A) The INTS2-MED13 rearrangement on Chromosome 17 is identified in 9 individuals and arises from an amplification. A tandem duplication that affects the 3' end of MED13 and the 5' end of INTS2 will fuse the promoter region of INTS2 to MED13. (B) The PPP1R9A-PSMC2 rearrangement on Chromosome 7 is identified in 6 individuals and arises from a deletion.

The phosphatase PTPN12 appears highly rearranged in 16 GBM patients, and it is a partner in a surprisingly large fraction (11/33) of the inferred fusion gene (Table 2.3). PTPN12 is known to dephosphorylate oncogenes c-ABL and Src; thus deregulation of PTPN12 might contribute to tumor survival [88]. While the 5' end of PTPN12 appears amplified with respect to the $\log_2$ copy number ratios at the 3' end, many inferred fusion genes consist of a deletion of the 3' end (i.e. Figure 2.12a). Additionally, some fusion gene candidates might indicate multiple rearrangements, such as a translocation occurring after an amplification that results in a fusion gene configuration (Figure 2.12b). Due to the large number of candidate rearrangement partners of PTPN12, it might be the deregulation of PTPN12, and not necessarily any single rearrangement, that is important for GBM.

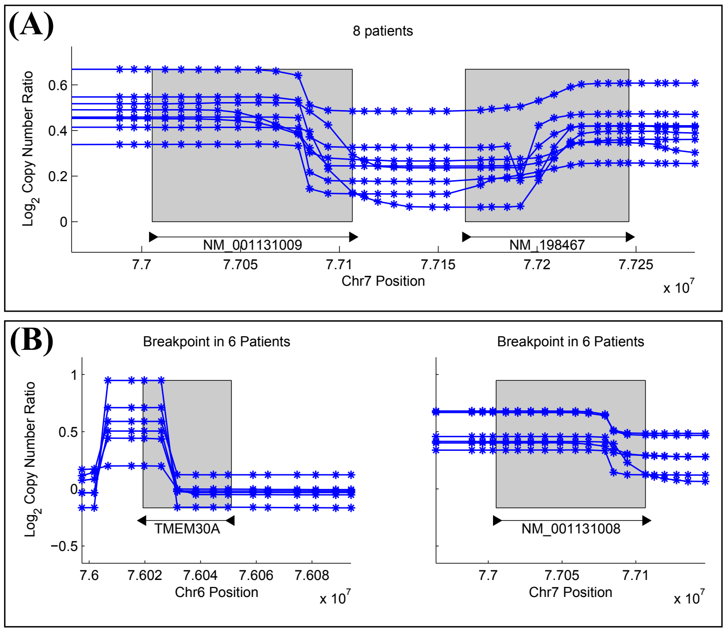

Figure 2.12: **Predicted Fusion Genes with PTPN12 as a Gene Partner.** (A) The inferred intrachromosomal fusion gene PTPN12/RSBN1L is one of two inferred intrachromosomal fusion genes. This fusion gene arises from a deletion within an amplified region, and is only present in 8 individuals out of 16 that have some rearrangement with PTPN12. (B) The inferred interchromosomal fusion gene TMEM30A-PTPN12 is one of 8 inferred interchromsomal fusion genes. While the breakpoint in TMEM30A appears to arise due to a short amplification, a translocation occurring after an amplification (where all of TMEM30A is amplified) may also explain this fusion gene signature.

Recurrent Gene PTPN12					
Gene		Genomic Location		# Patients	
PTPN12		chr7:77004708-77106533		16	
Inferred Intrachromosomal Fusion Genes					
5' End Gene		3' End Gene		# Patients	RMS
PTPN12	chr7:77005287-77106533	RSBN1L	chr7:77163678-77246421	8	0.1081
PTPN12	chr7:77004708-77106533	LUC7L2	chr7:138695173-138757626	8	0.2605
Inferred Interchromosomal Fusion Genes					
5' End Gene		3' End Gene		# Patients	RMSE
TMEM30A	chr6:76019357-76051074	PTPN12	chr7:77005287-77106533	6	0.1306
RNF150	chr4:142006174-142273412	PTPN12	chr7:77005287-77106533	5	0.1409
PTPN12	chr7:77005287-77106533	MED13	chr17:57374747-57497348	9	0.1906
CLK1	chr2:201425977-201434830	PTPN12	chr7:77005287-77106533	8	0.3168
ZRANB2	chr1:71301561-71319266	PTPN12	chr7:77005287-77106533	9	0.3250
PTPN12	chr7:77005287-77106533	UBR1	chr15:41022389-41185512	9	0.3475
PTPN12	chr7:77005287-77106533	LINGO1	chr15:75692423-75711712	8	0.3787
PPIL3	chr2:201443923-201460583	PTPN12	chr7:77004708-77106533	6	0.4741

Table 2.3: **Inferred Rearrangments involving PTPN12 in GBM.** The phosphatase PTPN12 appears in 10 inferred fusion genes, and is also an inferred gene truncation for 16 patients. The inferred rearrangements are ranked according to the root mean squared difference (RMS) of the copy number on either side of the fusion point (§2.2.4).

2.4 Discussion

We have introduced Neighborhood Breakpoint Correlation (NBC), an algorithm that identifies recurrent breakpoints in data from multiple individuals. NBC correctly identifies a known fusion gene (TMPRSS2-ERG) in aCGH data from 36 prostate tumors and infers gene truncations, structural variants, and fusion genes in aCGH data from glioblastoma. We expect that application of our method to additional samples will allow us to uncover and categorize other recurrent germline and somatic rearrangements.

NBC computes the probability that a breakpoint occurs between each pair of adjacent probes over *all* possible segmentations of a single copy number profile and then combines these probabilities across multiple profiles to identify recurrent breakpoints. The probabilistic approach contrasts with the typical methods for aCGH analysis that compute only a single segmentation of a copy number profile. Consideration of a single segmentation is reasonable for identifying recurrent aberrations because large aberrations will typically overlap in different individuals as long as the segmentations reasonably approximate the true underlying copy number level. However, identification of recurrent breakpoints is more sensitive to the choice of segmentation. Due to experimental noise in individual probes, the optimal segmentation of each individual profile may not "align" across profiles. Thus it is necessary to consider multiple suboptimal segmentations. Moreover the probabilistic model allows use to account for biological variability in the location of a breakpoint within a gene or other locus.

NBC successfully identifies known fusion genes and structural variants. For fusion genes, NBC's consideration of uncertainty and variability in the locations of breakpoints provides an advantage over methods that compare individual segmentations of copy number profiles. This advantage is mitigated for variants with highly conserved breakpoints such as germline structural variants that are common in a population. However, it is possible that NBC would be helpful for complex, or overlapping, structural variants, where recurrent breakpoints might be a stronger signal than recurrent aberrant intervals.

Additionally, we note that NBC is equally applicable to copy number profiles generated by mapping DNA sequence reads to a reference genome [28, 157]. With next generation sequencing technologies, breakpoint resolution can be much higher than most current aCGH methods, but the problems of breakpoint variability and uncertainty remain. Applying NBC to DNA sequencing data from multiple individuals will provide a more locally refined set of breakpoints; however, the number of sequenced cancer patients must grow before NBC is statistically powerful.

Chapter 3

SV Detection using Strobe Sequencing: A Parsimony Approach

We now move from array-based methods for CNV detection to sequence-based methods for SV detection. In particular, paired-end sequencing has improved the detection of SVs in human genomes by using a whole-genome resequencing approach (see §1.2).

In this chapter, we introduce an algorithm to identify and characterize structural variants with strobe reads by considering multiple possible alignments for each subread. The work in this chapter is taken from [124], and was originally presented at the Conference on High-Throughput Sequencing Methods and Applications (HitSeq) in 2010. We formulate the combinatorial optimization problem of selecting an alignment for each subread of every strobe read so that the total number of structural variants in the test genome is minimized. This generalizes a formulation that has proved successful for paired read analysis [52]. We show how to reduce the problem to an optimization problem on directed graphs, and derive an integer linear program (ILP) for the problem. We apply our method to simulated strobe sequencing data. We find that strobe reads outperform paired reads for SV detection. In particular, at a fixed sensitivity level strobe reads have nearly double the specificity of paired reads.

3.1 Related Work

Whole-genome sequencing for SV detection has been demonstrated for a handful of human genomes using different sequencing technologies. Older clone-based sequencing produced 2-150Kb fragments with 500bp-1Kb sequenced reads in phenotypically normal genomes [62, 143] and cancer genomes [120, 147]. Next-generation sequencing produces 200bp-3Kb fragments with 35-100bp sequenced reads [66].

Initially, computational methods to detect SVs from paired-end data simplified the issue of repeats by ignoring any paired-end fragment that aligned to a repetitive region of the genome. Methods such as GASV [133], PEMer [65], and Breakdancer [26] either removed paired reads with ambiguous mappings or chose the best mapping, breaking ties arbitrarily. Recently, the incorporation of ambiguous alignments for paired reads, have been found to improve the detection of SVs. VariationHunter [52, 53], GASVPro [134], and a method by [72] use probabilistic methods to choose a set of SVs from a list of clustered candidates by ensuring that each fragment supports a single SV in the set.

Unfortunately, none of the above methods are generalizable to the t-strobe framework. While they all work on 2-strobes, the multiply-linked subreads introduce a dependency on the selection of the discordant pairs associated with each strobe. To our knowledge, the method introduced in this chapter is the first published method for SV detection with strobe sequencing data.

Finally, we note that there are other methods to measure SVs from sequencing data besides the paired-end paradigm we have described here. One such method to measure SVs from sequencing data uses read coverage, which can be applied to single read data as well as paired-end read data. In these methods, copy number variants are determined by regions where the number of fragments that cover that region is different than the expected number of fragments [153, 157]. However, this can only measure CNVs and cannot identify copy-neutral variants. Another method for SV detection from sequencing data is to identify the reads that *span* the SV breakpoints in the test genome, which result in a gapped alignment in the reference [155]. These methods have the ability to identify the breakpoint location at the base pair resolution, but the repetitive nature of the reference genome often prohibits the identification of correct split reads. Finally, assembly-based methods use a *de novo* approach to assemble potential SVs that are particularly difficult to detect using a resequencing approach; these methods are often used as a post-processing step in the resequencing pipeline [63, 119].

3.2 Preliminaries

Here we formalize the basic generative model of structural variation that we will assume throughout the rest of this dissertation. This generative model has been described in detail in [133] for paired-end reads, and is generalized here for strobes. We model a test genome that is generated by independently adding, duplicating, removing, and rearranging segments of a reference genome. Each pair of adjacent coordinates in the test genome that are not adjacent in the reference genome is called an SV. The number of SVs in the test genome corresponds to the number of rearranged segments, and we assume that this number is relatively small given the size of the genome.

The test genome and the reference genome are strings from an alphabet Σ (typically, $\Sigma = \{A, C, G, T\}$). We model double-stranded DNA as two strings, a forward string and a reverse-complement string. The sequencing process is modeled by sampling substrings (from either strand) from the test genome and observing some parts of those substrings with an additional sequencing error inserted into the substring, in the form of substitution errors, insertion errors, or deletion errors.

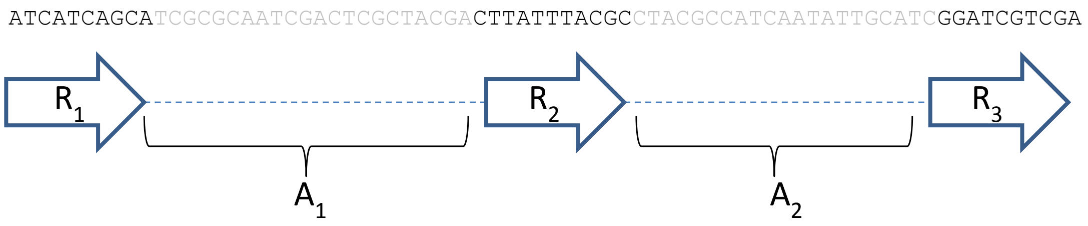
ATCATCAGCATCGCGCAATCGACTCGCTACGACTTATTTACGCCTACGCCATCAATATTGCATCGGATCGTCGA
R1
R2
R3
A1
A2

Figure 3.1: **Strobe Reads.** Here we show a t-strobe, where t=3. The subreads R_1,R_2, and R_3 are separated by strings of unknown bases A_1 and A_2 called advances. The subread length and advance length is determined by the amount of time the laser that reads the pulses of fluorescence is turned on and off.

Generative Model for Strobe Sequencing As mentioned in §1.2.1, a *strobe* (also called a t-strobe) is an alternating sequence of t *subreads*, whose sequence is observed, and $(t-1)$ *advances*, unknown sequences of "dark" bases. More formally, a strobe S is comprised of a set of t subreads $\{R_1^{(S)}, R_2^{(S)}, \ldots, R_t^{(S)}\}$ (Figure 3.1). The subreads are strings from Σ, and have variable length distributed around some mean.

Each subread $R_i^{(S)}$ has a set of *subread alignments* $A(R_i^{(S)})$ to the reference genome. There are three pieces of information associated with each subread alignment $a \in A(R_i^{(S)})$: the interval $[x_a, y_a]$ in the reference genome corresponding to the alignment location (where $x_a < y_a$), the orientation $sign_a \in \{F, R\}$ of the alignment that determines whether the alignment is on the forward or reverse-complement strand, and the number of errors ϵ_a (mismatches, insertions, and deletions) in the alignment.

Inferring SVs from Strobes We define a *consecutive subread pair* as a pair of subread alignments from adjacent subreads in the same strobe S, i.e. $a_1 \in A(R_i^{(S)})$ and $a_2 \in A(R_{i+1}^{(S)})$ for $i < t$. The consecutive subread pair (a_1, a_2) is *concordant* if the aligned distance and orientation of the pair is expected given the generative model; that is, they have the following constraints:

1. $sign_{a_1} = sign_{a_2}$ (they are in the same orientation in the reference).
2. If $sign_{a_1} = sign_{a_2} = F$, then $l \leq x_{a_2} - y_{a_1} \leq u$ for bounds l and u on the advance length.
3. If $sign_{a_1} = sign_{a_2} = R$, then $l \leq x_{a_1} - y_{a_2} \leq u$ for bounds l and u on the advance length.

The lower l and upper u bounds are bounds on the advance length (the unknown sequence between subreads in a strobe). These bounds are often estimated from an empirical distribution of all consecutive subread pairs, since much of the sequence between the test and reference genomes are similar. If the consecutive subread pair (a_1, a_2) is concordant, then this suggests that there is no difference (in terms of large structural variants) between the reference genome and the test genome within the interval defined by the subread alignments to the reference.

If the consecutive subread pair (a_1, a_2) is not concordant, then it is *discordant* and implies an SV in the test genome. Further, these discordant pairs are categorized according to how they are unexpected, either in terms of the orientation or the distance between the subread alignments, called the *aligned length*. Some examples of SVs and the type of discordant pairs that imply them are the following:

1. If there is a deletion in the test genome compared to the reference genome, then there is extra sequence in the reference genome between the two subreads and thus the aligned length will be larger than expected.

2. If there is an insertion in the test genome compared to the reference genome, there is missing sequence in the reference and the aligned length will be smaller than expected.

3. If there is an inversion in the test genome compared to the reference genome, then the orientation of the two subread alignments will differ.

4. If there is a translocation in the test genome compared to the reference genome, then there is no notion of aligned length because the subread alignments are on different chromosomes.

An SV is defined by disparate coordinates in the reference genome that are near each other in the test genome. If a discordant pair (a_1, a_2) suggests a deletion, for example, the two coordinates (x, y) in the reference that denote the deletion satisfy the equation

$$l \leq (x - y_{a_1}) + (x_{a_2} - y) \leq u. \tag{3.1}$$

Similar equations hold for inversions, insertions, translocations and other variants [8] (Figure 3.2).

A collection of discordant pairs indicate the *same* SV, if they simultaneously satisfy these equations for a particular choice of cut points. Regardless of the number of cut points determined by a discordant pair (two in the case of deletions and one in the case of insertions, inversions, and translocations), we say that each discordant pair represents a single *breakpoint*. Thus, a collection of discordant pairs indicate the same breakpoint. Note that an SV may be defined as two cut points in the same breakpoint (as in a deletion), or as two cut points in two different breakpoints (as in a balanced inversion) (Figure 3.2). Breakpoints are only approximately defined according to the uncertainty in the advance lengths (e.g. according to Equation 3.1), meaning that there is some number of cut points for which the inequalities hold true.

Generative Model for Paired-End Sequencing The generative model for paired-end sequencing, which is described in [133], is that a fixed number of characters are observed from the *ends* of the substrings sampled from the test genome, on opposite strands, producing a pair of *reads*. These reads are aligned to the reference genome, and a selection of one alignment for each read produces an alignment pair, which is either concordant or discordant according to inequalities similar to those above but adjusted for the expected orientation of the read pairs.

Fundamentally, the difference between a strobe-sequenced substring and a paired-end-sequenced substring from the test genome is the expected orientation of the subreads and reads (Figure 1.4). Note that if

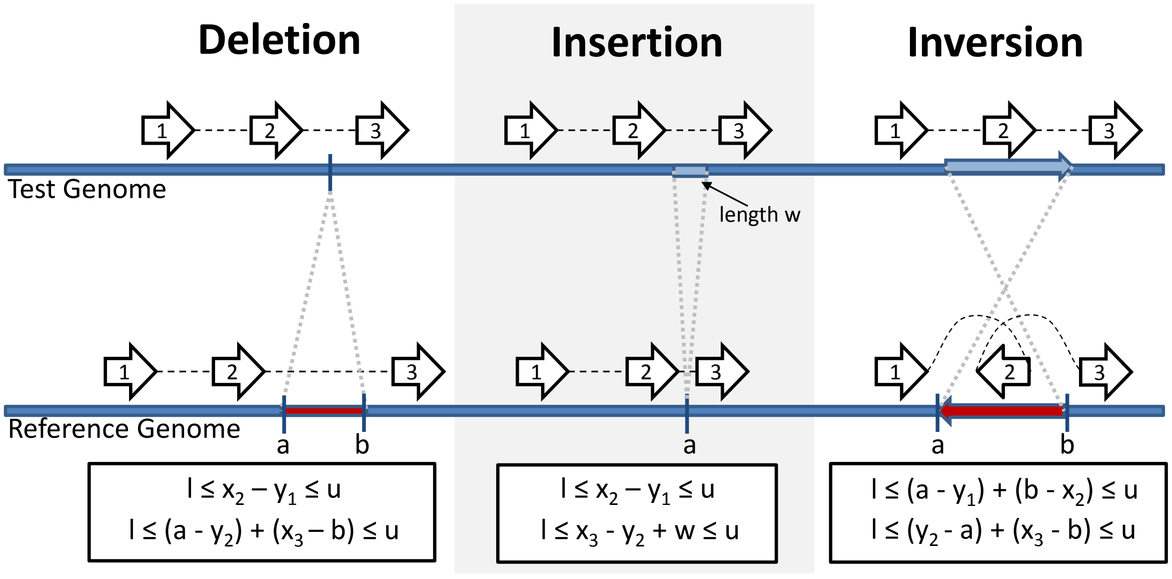

Figure 3.2: **Discordant pairs that imply SVs and the constraints they pose on the type of variant.** For simplicity, we label the subreads 1, 2 and 3, and the corresponding intervals as $[x_1, y_1]$, $[x_2, y_2]$, and $[x_3, y_3]$. SV coordinates are labeled as a and b in the reference genome. Each strobe corresponds to two inequalities due to two sets of adjacent subreads $(1, 2)$ and $(1, 3)$. In the deletion and insertion example, the set of adjacent subreads $(1, 2)$ results in a concordant pair. All other inequalities represent discordant pairs. Note that insertions are a special case that includes the length of the inserted block rather than two cut points. The inequalities for an example translocation are not shown.

we invert the right-most read for paired reads, we have a variant of a 2-strobe. Thus, we can treat alignment pairs from paired-end sequencing as 2-strobes with *fixed* subread lengths (as opposed to variable subread lengths), different bounds on the advance lengths (determined by the empirical distribution of the sampled substrings), and a different model for sequence errors (substitutions, insertions, and deletions).

3.3 A Combinatorial Optimization Method for SV Detection with Strobes

For each strobe S, the selection of an alignment $a \in A(R_i^{(S)})$ fixes the location of $R_i^{(S)}$, and choosing an alignment for every subread in S will produce a set of consecutive subread pairs. Because the test genome is sequenced to a particular coverage, our intuition is that breakpoints of true structural variants will be supported by many strobes. Thus, we aim to choose an alignment for each subread of every strobe so that the resulting set of structural variants is *optimal* according to some objective function. This problem has been considered in the paired read case by [52, 72]. In particular, [52] defines the maximum parsimony objective function of choosing alignments to *minimize* the number of inferred structural variants. Below we consider the equivalent problem for strobe reads, show a reduction to an optimization problem on directed graphs, and derive an integer linear program (ILP) for the problem.

3.3.1 Problem Formulation

An alignment for strobe S is obtained by selecting an alignment $a \in A(R_i^{(S)})$ for each subread $R_i^{(S)}$. Let $A(S) = A(R_1^{(S)}) \times A(R_2^{(S)}) \times \cdots \times A(R_{t-1}^{(S)})$ be the set of alignments for S. For $a \in A(S)$, let $B(a)$ be the set of genomic breakpoints indicated by a. For ease of exposition assume that all discordant consecutive subread pairs are deletions: $B(a)$ is then the set of coordinates (x, y) that satisfy Equation 3.1 for the discordant pairs in a. If a consists entirely of concordant pairs, then $B(a) = \emptyset$. We define the following problem we wish to solve.

t-Strobe Minimum Breakpoints Problem. Given a set $\mathbf{S}$ of n t-strobes and the corresponding strobe alignments $A(S)$ for each $S \in \mathbf{S}$, find a set of breakpoints $\mathcal{B}$ of minimum cardinality such that for all $S \in \mathbf{S}$ there is an $a \in A(S)$ with $B(a) \subseteq \mathcal{B}$.

This problem is NP-hard, as was shown for the paired read case ($t = 2$) by a reduction from the Set Cover problem [52]. Below, we reformulate this problem as an integer linear program (ILP). We derive the ILP from a directed graph $G = (V, E)$ that represents breakpoints shared by multiple strobes (Figure 3.3).

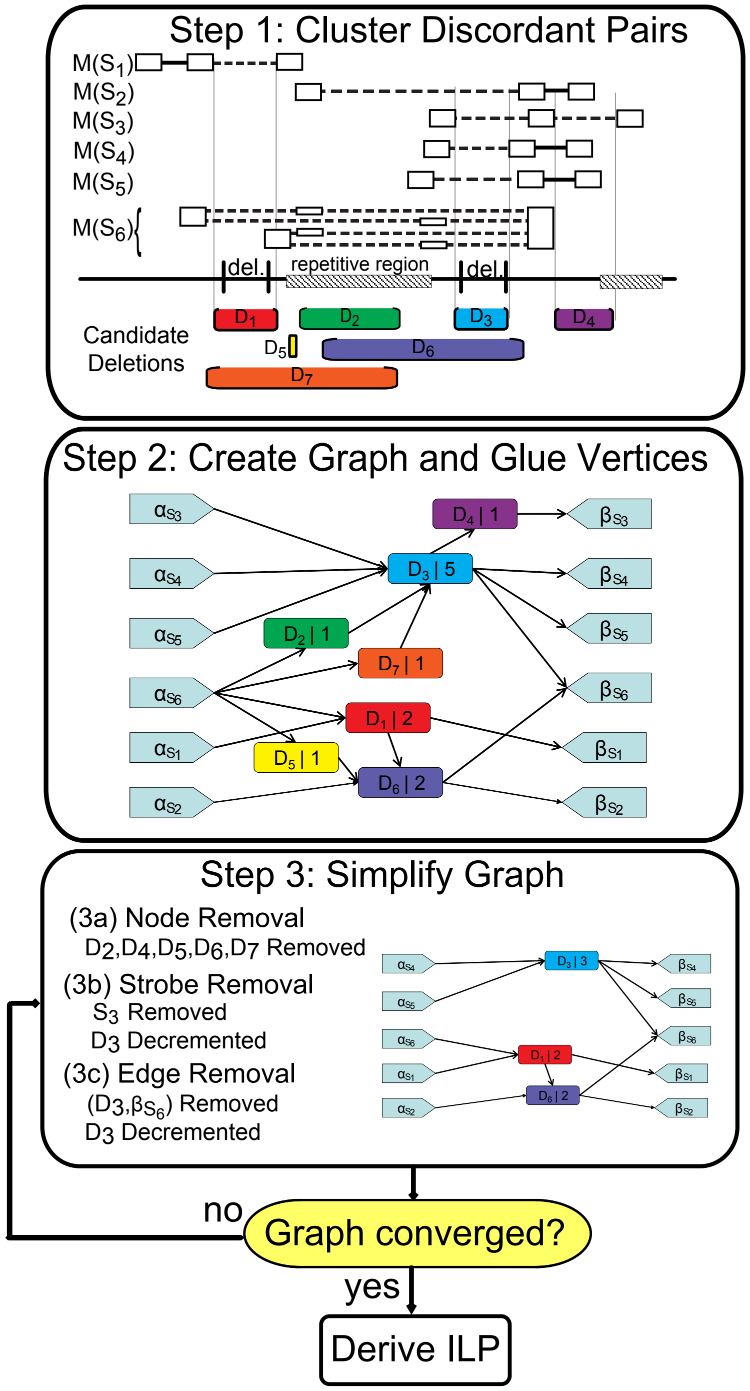

Figure 3.3: Construction of graph for 3-strobes $S_1, \ldots, S_6$. In step 1, the discordant pairs (dotted lines) representing a particular breakpoint type (deletions) are clustered, producing 7 candidate deletions. In step 2, a graph is created with a source vertex α and a sink vertex β for each strobe and one vertex for each cluster. The number on each vertex indicates the number of strobes with a discordant pair consistent with the deletion. In step 3, the graph is simplified using heuristics described in § 3.3.4. Step 3 is repeated until the graph has converged, upon which it is used as input to the ILP.

3.3.2 Graph Construction

We begin by constructing a graph whose edges are alignments of a subread and vertices are advances between subreads. Formally, consider an individual strobe read S. We represent the set of all possible alignments for S with a directed graph $G_S = (V_S, E_S)$. The vertex set

$$V_S = \bigcup_{a=1}^{t-1} \left(A(R_i^{(S)}) \times A(R_{i+1}^{(S)}) \right) \bigcup \alpha_S \bigcup \beta_S \tag{3.2}$$

is the set of all possible consecutive subread pairs, with an additional source vertex α_S and sink vertex β_S corresponding to the start of the first subread and the end of the last subread, respectively. We refer to vertices that are not sources or sinks as *internal* vertices. The edge set E_S consists of three types of edges:

1. Edges from the source vertex α_S to alignments from first subread:

$$\bigcup_{\substack{v=(a_1,a_2):\\ a_1 \in A(R_1^{(S)})}} (\alpha_S, v)\,. \tag{3.3}$$

2. Edges from the alignments from the last subread to the sink vertex β_S:

$$\bigcup_{\substack{v=(a_1,a_2):\\ a_2 \in A(R_t^{(S)})}} (v, \beta_S)\,. \tag{3.4}$$

3. Edges between internal vertices:

$$\bigcup_{\substack{u=(a_1,a_2),\\ v=(a_2,a_3)}} (u, v)\,. \tag{3.5}$$

The edges E_S simply connects vertices that represent the same choice of alignment. The alignments for strobe S are exactly the set of paths in G_S from α_S to β_S, where the vertices in the path represent the selection of alignments $a \in A(R_i^{(S)})$ for each subread $R_i^{(S)}$. We anticipate that strobe alignments consisting of completely concordant pairs are more often correct than strobe alignments consisting of some discordant pairs. If there exists a choice of alignments that results in a concordant pair for adjacent subreads, then we ignore all other alignment choices with a lower alignment score. Note that this does not imply that all concordant pairs must be chosen, but rather that if a discordant pair is chosen instead of a concordant pair then at least the alignment score for the alignments in the discordant pair are better than score for the alignments in the concordant pair. In the graph, if an internal vertex v represents a concordant pair then for every pair of edges (u, v) and (v, w) we add an edge (u, w) and remove v from the graph. The new edges (u, w) include a selection of alignments corresponding to the original vertex v.

Now we form a graph $G = (V, E)$ by merging vertices in the graphs G_S whose alignments are consistent with a single set of breakpoint according to an inequality such as (3.1). We compute the vertices to merge using GASV [133], a program that efficiently computes whether paired reads indicate the same breakpoint using a computational geometry algorithm. In each merged vertex, we store the identities of subread alignments from the original vertices. Note that G may be a multigraph. The t-Strobe Minimum Breakpoints Problem reduces to finding a subgraph H of minimum cardinality such that H contains a path from source α_S to sink β_S for each strobe $S \in \mathbf{S}$. Note that H will always contain the source and sink vertices, so the minimum cardinality is at least $2n$.

3.3.3 Integer Linear Program Formulation

The graph formulation is suggestive of a fixed charge multi-commodity network flow problem [35], with each strobe representing a distinct commodity. Briefly, given a directed graph with capacities and costs on each edge and a set of demands to satisfy between source vertices and sink vertices, the fixed charge multi-commodity network flow problem aims to satisfy the demands while minimizing some cost. This corresponds to assigning a demand of one to each source node α_S and expecting a demand of one to be satisfied for each sink node β_S while placing costs on vertices.

However, our problem differs from this and related problems in that we require a path from α_S to β_S. Simply using net flow does not capture this information (Figure 3.4). Thus, we need to maintain separate accounting of each strobe entering and exiting a vertex rather than merely accounting for the net flow as in a multi-commodity flow problem.

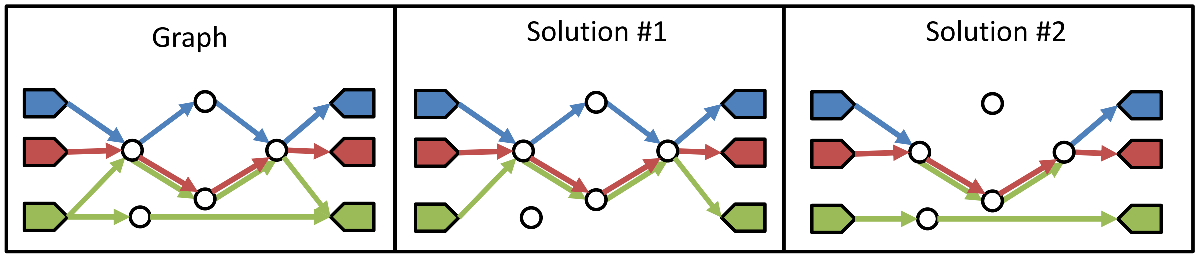

Figure 3.4: **The graph G that is constructed is not an example of a fixed-charge network flow algorithm.** Here we show an example graph G and potential solutions to the fixed-charge network-flow problem. The graph edges are colored by the three strobes (blue, red and green) that are allowed to travel that edge. Solution #1 shows a feasible solution to the fixed-charged network-flow problem uses four internal vertices. Solution #2 also contains with four internal vertices. However, solution #2 does not result in a path from source to sink for each individual strobe. Thus, solution #2 is not a feasible solution to the t-Strobe Minimum Breakpoints Problem.

Motivated by an ILP for the fixed charge flow problem [51], we formulate our problem as an ILP. For each vertex $v \in G$ we define binary indicator variables p_v such that $p_v = 1$ if and only if v is in the optimal solution. Similarly, we introduce variables $q^{(S)}_{(u,v)}$, which represent the flow across edge (u, v) for strobe

$S \in \mathbf{S}$. Lastly, we define $N_{v+}^{(S)}$ and $N_{v-}^{(S)}$ as the outward and inward neighbors, respectively, of vertex v and strobe $S \in \mathbf{S}$. Our integer linear program for minimizing the total number of SVs is

$$\min \sum_{v \in V} p_v \tag{3.6}$$

$$s.t.$$

$$\begin{aligned} & p_v \in \{0,1\} && \forall\, v \in V \\ & 0 \leq q_{(u,v)}^{(S)} \leq p_u && \forall\, (u,v) \in V,\ S \in \mathbf{S} \\ & 0 \leq q_{(u,v)}^{(S)} \leq p_v && \forall\, (u,v) \in V,\ S \in \mathbf{S} \end{aligned} \tag{3.7}$$

$$\sum_{j \in N_{v+}^{(S)}} q_{(u,v)}^{(S)} - \sum_{j N_{v-}^{(S)}} q_{(v,u)}^{(S)} = \begin{cases} 1 & \text{if } v = \alpha_S \\ -1 & \text{if } v = \beta_S \\ 0 & \text{otherwise.} \end{cases} \tag{3.8}$$

Recall that each vertex v (except for the α and β vertices) corresponds to a breakpoint. The objective (3.6), minimizes the number of vertices, thus minimizing the number of breakpoints. In the optimal solution, the flow of each edge $q_{(v,u)}^{(S)}$ must have weight 0 or 1 (i.e. it is either used once for strobe S or not used at all). Note that the flow for any edge,

$$q_{(u,v)} = \sum_{S \in \mathbf{S}} q_{(v,u)}^{(S)}, \tag{3.9}$$

is only non-zero if u and v are in the optimal solution. Constraint (3.8) ensures that each strobe has a valid sequence of subread alignments. A solution of the ILP corresponds to a selection of internal vertices from G.

3.3.4 Graph Simplification

In order to improve performance, we developed several heuristics to simplify the graph G before solving the ILP (Figure 3.3).

1. **Vertex removal** We define the *support* of a vertex as the number of strobe reads with paths through it. We require that a vertex be supported by at least Δ strobes; thus we remove internal vertices (and their adjacent edges) with support less than Δ. Note that we must count the number of strobe reads that travel through the vertex, rather than the number of discordant pairs that cluster together, because the same strobe might have multiple discordant pairs supporting the same breakpoint.

2. **Strobe removal** Following vertex removal, some strobes may no longer have a path from source to sink. We remove such strobes from the graph. We use a dynamic program to efficiently check that at least one path exists for each strobe S.

3. **Edge removal** Following vertex removal, each strobe S will have at least one path from source to sink, but may have extraneous alignments that do not lie on any path from source to sink. We remove such edges from the graph, since they cannot appear in the ILP solution due to the flow constraint. We use a dynamic program to efficiently find all alignments that do not lie on any path from source to sink for each strobe S_i.

We iteratively perform these three operations on G until no more vertices, strobe reads, or edges are removed. We then use this modified graph as input to the ILP described above and run it on an ILP solver.

3.4 Results

We applied our algorithm to simulated strobe sequencing data from Pacific Biosciences. We first generated two datasets from the reference genome by introducing a set of rearrangements, and then from these datasets we simulated strobes. We analyze two datasets: structural variants identified in the Venter whole-genome assembly [73] and a synthetic complex rearrangement in highly repetitive regions.

3.4.1 Datasets

We simulated two datasets and assessed the detectability of deletions, inversions, and translocations from simulated strobe and paired-end datasets.

Venter Dataset We simulated a test chromosome based on known rearrangements from Chr17 of the Venter genome [73], following the procedure presented in [26]. Given a list of 17,376 insertions, deletions, and inversions on Chr17, we concatenated intervals from the hg18 human reference corresponding to the rearrangements. We simulated 3Kb strobes containing three 200bp subreads and two 1200bp advances at 10x, 20x and 30x coverage, producing 262,355, 524,709, and 787,064 strobes, respectively.

We then introduced error into each subread using PacBio's error simulator. When these results were first published, the capabilities of the Pacific Biosciences commercial machine were not yet known since data first became available in early 2010. Thus, we assumed a sequencing error rate of 5%. We used Pacific Biosciences' error simulator, to accurately model the errors in their single-molecule sequencing technology. Specifically, the simulator models the higher rate of insertions and deletions (using a a roughly equivalent ratio of each) relative to miscall errors in subreads that is typically seen in their data [39].

We aligned the subreads to Chr17 of hg18 using PacBio's in-house aligner, BLASR. BLASR is designed to quickly align large reads and is tolerant to a wide range of sequencing errors [20].

Repetitive Dataset Repetitive regions in the genome are notoriously difficult for structural variant detection. To test the ability of strobe reads to capture breakpoints near repetitive regions, we constructed a 11.6kb sequence with two translocations by concatenating three different transposons from hg18: a 6kb L1-family

LINE (chr2:181406133-181413161), a 503bp Alu (chr7:66854543-66856104), and a 3kb L2-family LINE (chr15:87930634-87933678), each flanked by 500 basepairs. From this sequence, we generated 10 simulations with 10x coverage using the same sized 3Kb strobe reads as above and introduced 5% sequencing error. We aligned the subreads to the entire hg18 genome using BLASR.

Comparison to Paired Reads For each dataset, we compare the results with strobe to those obtained via paired reads. To remove differences due to read alignment, we simulate paired reads explicitly using the strobe datasets. We consider two sets of paired reads by transforming each t-strobe S:

1. **Paired Read Library (1.6Kb fragment length)** We define the set of paired reads that are adjacent subreads of S,
$$\text{Paired Read Library } = \bigcup_{i=1}^{t-1} \left(A(R_i^{(S)}) \times A(R_{i+1}^{(S)}) \right). \tag{3.10}$$

2. **Mixed Paired Read Library (1.6Kb & 3Kb fragment lengths)** We define the set of paired reads that are all combinations of subread pairs of S,
$$\text{Mixed Paired Read Library } = \bigcup_{1 \leq i < j < t} \left(A(R_i^{(S)}) \times A(R_j^{(S)}) \right). \tag{3.11}$$

The Paired Read Library dataset corresponds to a paired read dataset generated by a single size selection, while the Mixed Paired Read Library dataset corresponds to multiple fragment sizes. Note that for Illumina and ABI SOLiD machines, the latter requires preparation of multiple sequencing libraries. Given a set of strobe reads with physical coverage c, the Paired Read Library will have approximately the same physical coverage as strobe reads with twice as many reads. The Mixed Paired Read Library will have physical coverage $2c$ with three times as many reads. Thus, we subsample the reads in the Mixed Paired Read Library to achieve physical coverage c.

3.4.2 Variant Detection on the Venter Dataset

We tested our method on 124 deletions and four inversions from the Venter chromosome. We first present the computations used to assess sensitivity and specificity in detecting a set of known deletions. We then report the sensitivity and sensitivity of strobes and mate-pairs by varying Δ, the minimum support of a vertex in the graph G. Finally, we report the results for the four inversions.

ROC Computations for Deletions

Consider a set of *true* deletions defined by intervals $T = \{[a_1, b_1], \ldots, [a_{|T|}, b_{|T|}]\}$ known to be present in the test genome. Given a set $P = \{[c_1, d_1], \ldots, [c_{|P|}, d_{|P|}]\}$ of *predicted* (or inferred) deletions from an ILP solution, we compute two different ROC-type plots.

1. **Variant-based "ROC"** We wish to count the number of real deletions that the ILP infers; however we must be careful in how we count because the inferred deletions are approximate due to the advance length bounds. Thus, we count the number of true deletions $[a_i, b_i] \in T$ that have a nonempty intersection with some interval $[c_j, d_j] \in P$ and

$$(d_j - c_j) \leq (b_i - a_i) + u,$$

where u is a the upper bound for the advance lengths and can be estimated from the collection of advance length bounds. The equation above ensures that length of the inferred deletion length must be smaller than the true deletion length plus the largest allowed length of an advance. Let the number of such deletions be V. The true positive rate is $\frac{V}{|P|}$, and the number of false positives $|P| - V$. Note that this is not a true ROC curve because we do not compute a false positive rate, but rather report the number of false positives (hence the quotes).

2. **Pair-based ROC.** It is possible that, while we may have incorrectly predicted the deletion intervals, we might have correctly identified which discordant pairs support true deletions. For example, nearby deletions might produce incorrect breakpoint locations but contain the expected discordant pairs. Thus, we also count the number of discordant pairs that support some true deletion. Let $D(T)$ be the set of discordant pairs that support some true deletion in T, and let $\overline{D(T)}$ be the remaining discordant pairs. Similarly, let $D(P)$ be the discordant pairs that support inferred deletions in P. The true positive rate (TPR) and false positive rate (FPR) are then the following:

$$\text{TPR} = \frac{|D(P) \cap D(T)|}{|D(T)|} \text{ and FPR } = \frac{|D(P) \cap \overline{D(T)}|}{|\overline{D(T)}|}.$$

Deletions

We computed the variant-based and pair-based ROC curves for the selected set of 124 deletions greater than 120 basepairs, (Figure 3.5). For the variant-based ROC, strobes outperform paired reads for all three coverages in 'Area under the ROC Curve' (AROC) values. Moreover, strobes outperform Mixed Paired Reads for 10x and 30x coverages, where the advantage in AROC for the Mixed Paired Read library is a result of slightly better specificity at extremely low ($< 20\%$) values of sensitivity. On average, at fixed values of sensitivity, strobe reads make $57.18\% \pm 4.282$ fewer false positives than paired reads. At 20x coverage and a maximum sensitivity of 87.10% for strobe reads, 90.32% for the Paired Read library, and 92.74% for the Mixed Paired Read library, strobe reads make 45.13% fewer false positives than the Paired Read library, and 61.53% fewer false positives than the Mixed Paired Read library.

As noted, it is possible that some of the false positive breakpoints contain true discordant pairs whose breakpoint intervals are incorrectly predicted. 39 of the considered deletions have another event (at least

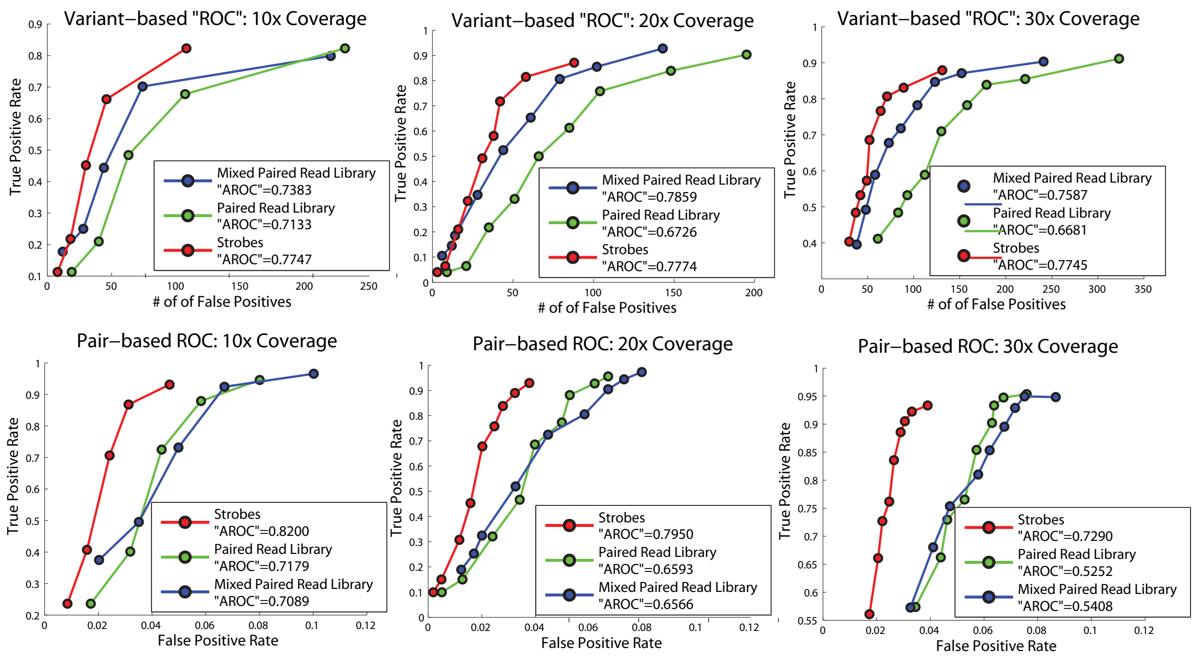

Figure 3.5: **ROC curves for the Venter simulation.** The top row is a variant-based "ROC" curve for 124 deletions $\geq$ 120 basepairs, which computes the number of real deletions out of 124 that appear in the solution. The bottom row is a pair-based ROC curve for the same 124 deletions, which counts the number of discordant pairs that support the real deletions. The reported "AROC" is the area under the curves normalized by the maximum x-value in each plot. For 10x coverage, Δ ranges from 2 to 10 in steps of 2. For 20x and 30x coverage Δ ranges from 4 to 20 in steps of 2.

50 bp in size) within the corresponding advance which may affect the breakpoint locations. Thus, the pair-based ROC curve quantifies the number of discordant pairs that are correct independent of the determined breakpoint locations. In the pair-based ROC curves, strobes outperform paired reads and mixed paired reads for all three coverages in terms of AROC values. Strobes decrease the false positive rate by an average of $50.83\% \pm 4.83$ compared to the Paired Read datasets and $56.07\% \pm 8.11$ compared to the Mixed Paired Read libraries at fixed sensitivity.

Inversions

Only four inversions appear on Venter Chromosome 17 with lengths detectable by the simulated sequencing data. Here, the true positive rate computation described previously for deletions does not apply because the inequality assumes that a region is lost, while the region size isn't affected by an inversion. Instead, we counted an inferred inversion as a true positive if the mutual intersection between the inversion interval and the prediction interval was greater than 50%, following [26]. Our method detects the two longest inversions of these four for the Strobe and Paired Read libraries, while the Strobe dataset infers 22% fewer false positives (Table 3.1).

Left Breakpoint Coordinate	Length	Strobe	Paired Read	Mixed Paired Read
5,826,739	552bp			
40,566,233	1,151bp	✓	✓	
55,552,838	3,557bp	✓	✓	✓
57,999,778	472bp			
Total # Predicted Inversions:		23	96	52

Table 3.1: **Detection of Inversions from the Venter Simulation.** Results from the $20X$ coverage Strobe dataset, Paired Read dataset, and Mixed paired read dataset with $\Delta = 10$. The Strobe and Paired Read libraries detect 2 of the 4 inversions, while the Mixed Paired Read library detects only the longest inversion.

Graph Analysis of the Venter Dataset

The size and topology of the graph G used as input to the ILP varies between the Strobe, Paired Read and Mixed Paired Read libraries, affecting the runtime of the integer linear program. We investigated this for the graphs constructed for the set of deletions described above with minimum support $\Delta = 8$ (Table 3.2). We found that the strobe datasets construct graphs with fewer edges and vertices than the paired read and mixed paired read libraries. For example, with support $\Delta = 8$ there are approximately 65% fewer edges and internal vertices for 10x coverage, approximately 80% fewer edges and internal vertices for 20x coverage, and approximately 83% fewer edges and internal vertices for 30x coverage. Note that the strobe and paired read datasets predict the same number of clusters at each coverage (which is expected due to the paired read library construction), and the Mixed Paired Read library uses a smaller set of clusters to construct the graph G.

	Clustering Statistics			Graph Statistics				ILP Sol. Stats	
	#	#	# Clusters	# Strobes/Pairs		# Internal	#	# Verts that are	
Dataset*	Discord.	Del.	(≥ 8)	Removed	Retained	Verts	Edges	in Sol.	Correct
10x S	50,554	12,762	145	362	410	50	978	45	27
10x PR	50,543	12,762	145	321	518	143	2,638	66	26
10x MPR	35,881	5,984	83	330	542	83	1,650	59	31
20x S	101,367	25,701	807	389	1,585	159	4,182	131	89
20x PR	101352	25,698	807	265	1,974	785	19,270	198	94
20x MPR	70,022	11,896	372	288	1,976	365	8,658	179	100
30x S	151,053	38,711	1,492	545	2,540	229	7,784	171	100
30x PR	151034	38,705	1,492	345	3,240	1,465	42,322	283	104
30x MPR	107,577	19,451	810	430	3,028	802	21,650	228	105

*S=Strobe, PR=Paired Read, MPR=Mixed Paired Read.

Table 3.2: **Clustering, graph construction, and ILP solution statistics for the Venter simulation with** $\Delta = 8$. The clustering statistics include the number of discordant pairs after removing concordant alignments, the number of these discordant pairs that indicate deletions, and the number of clusters that contain 8 or more discordant pairs. The graph statistics include the number of strobe reads or paired ends that are removed from the graph, the number that are retained, the number of internal vertices, and the number of edges in the graph. The solution statistics report the number of internal vertices in the final ILP solution and the number of these vertices that are in the list of 124 deletions $\geq$ 120bp.

In addition to the size of the input graph G, the graph topology is different between the strobe datasets and the paired read libraries. In general, graphs with many connected components are easily parallelizable, as each connected component could be run independently of the others. While all graphs contain many connected components, the graphs of Paired Read and Mixed Paired Read libraries contain connected components with many more internal vertices than graphs with Strobe data. For example, at 10x coverage with $\Delta = 4$, the largest connected component for the Paired Read library contained 350 internal vertices while the largest connected component for the Strobe dataset contained only 6 internal vertices. One such connected component from the Strobe dataset with 10x coverage is shown in (Figure 3.6).

Comparison to Short Read Sequencing

We illustrate the advantages of the longer fragments and subreads of strobe sequencing by comparing strobe reads to simulated paired read data that approximates the fragments sizes and read lengths that are routinely obtained with short read, short insert sequencing technologies (Figure 3.7).

Using the same Venter Chr17, we simulated 200bp fragments with 50bp reads using the wgsim program from SAMtools [75] at 20x coverage. The simulated fragments have a base error rate of 0.02, a mutation rate of 0.001, 10% indels, and 30% probability that an indel is extended. We aligned the reads using BWA /gpfs/data/compbio/users/aritz/StrobeSV2/results/scripts. We then ran GASV [133] and VariationHunter [52] on the discordant pairs that map uniquely to the reference. Since VariationHunter utilizes reads with non-unique alignments, we considered discordant pairs that have multiple alignments to the reference. We

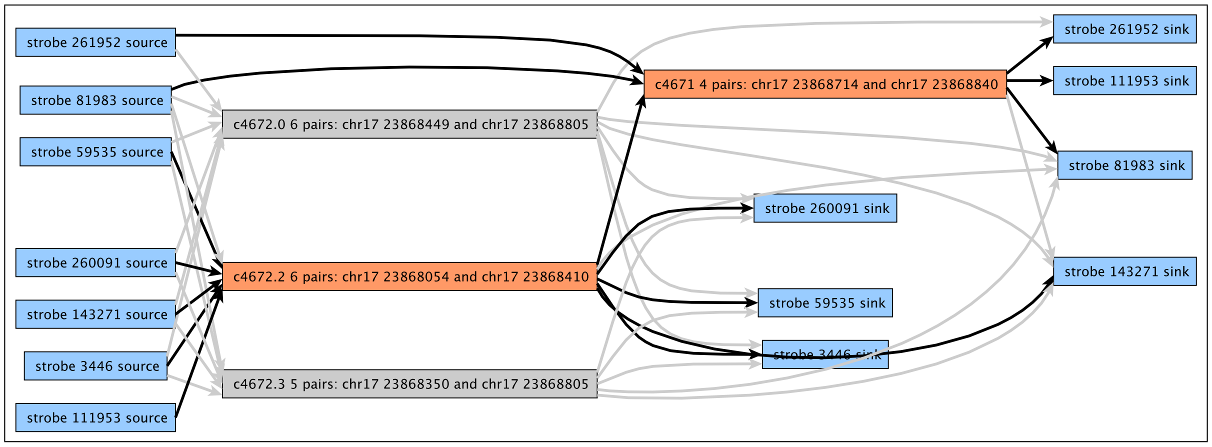

Figure 3.6: **An example of a connected component in the solution.** A connected component from the solution of the 10x coverage Strobe dataset with $\Delta = 4$. Edges in the final solution are in black. 6 strobe reads support the first variant in the solution, 4 strobe reads support the second variant in the solution, and two strobe reads span both variants.

considered reads with low mapping quality (≤ 10) for BWA and aligned them with Novoalign, an aligner that has higher sensitivity than BWA at the cost of a longer running time [68]. We considered up to 100 alignments for reads aligned with Novoalign. At maximum specificity for strobe reads, where the true positive rate is 0.87, VariationHunter achieves a true positive rate of 0.23 and GASV achieves a true positive rate of 0.34 at approximately the same sensitivity (between 85-95 false positives).

We emphasize that this comparison is limited by its use of simulated data, and by our use of VariationHunter and GASV without further post-processing. For example, the VariationHunter publication [52] describes several additional steps used to achieve better performance. Additionally, uncontrolled simulation parameters such as the fragment length and the subread length affect the comparison, and explicitly comparing the performance of different types of sequencing platforms is beyond the scope of this thesis. Since this is not an "apples-to-apples" comparison, the main motivation of this work is to demonstrate the ability of extra pairing information, *not* to demonstrate that our method applied to simulated strobe sequencing data is better than other methods applied to simulated paired read data.

3.4.3 Variant Detection for the Repeat Dataset

We now move from the Venter dataset to the Repeat dataset we have previously described. We make two points with this dataset: (1) our method can detect translocations, an SV that was missing in the Venter dataset due to the single chromosome simulation, and (2) our method can detect breakpoints in highly-repetitive sequences. We ran our method on the 10 strobe datasets, the 10 paired read datasets, and the 10 mixed paired read datasets. We computed the variant-based ROC by varying the support Δ and then averaging the values over the 10 simulations. The strobe datasets and Paired Read libraries report similar true positive rates on average, while the Strobe dataset reports fewer false positives (Figure 3.8). Many

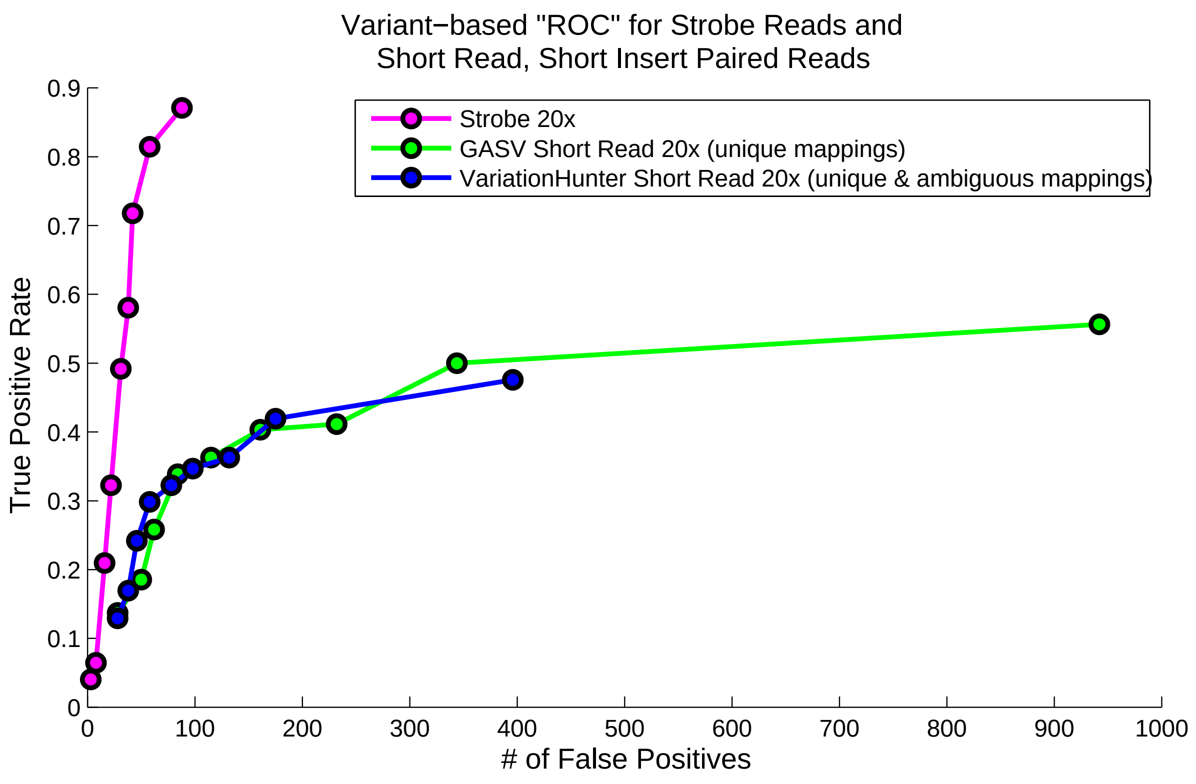

Figure 3.7: **Strobe reads vs. paired-end reads on the Venter simulation.** Δ ranges from 4 to 20 in steps of 2 for all curves.

subreads in the simulations align to hundreds of different regions on the genome, causing the Paired Read libraries to contain many internal vertices in the input graph for the ILP (Table 3.3). However, the number of vertices in the graph is greatly reduced using strobe reads, indicating that many pairs in these repetitive regions are eliminated by incorporating concordant information (Figure 3.8).

Dataset	Avg # Input Vertices	Avg # Final Inferred Translocations	Avg # Correct Inferred Translocations
Strobe	16 ± 25.88	2.1 ± 0.58	1.9 ± 0.32
Paired Read	85 ± 106.12	3.1 ± 1.37	1.9 ± 0.32
Mixed P.R	133 ± 183.20	2.9 ± 1.60	1.7 ± 0.48

Table 3.3: **Statistics from the Repeat Simulation.** For the Strobe dataset, the Paired Read dataset, and the Mixed Paired Read dataset, we note the number of internal vertices in the input graph G, the number of inferred translocations, and the number of correct inferred translocations out of a total of 2. These values are averaged over the 10 simulations for each dataset.

3.5 Discussion

We have introduced a combinatorial algorithm for structural variant identification from strobe sequencing data. This parsimony-based optimization objective to minimize the total number of SVs inferred by the

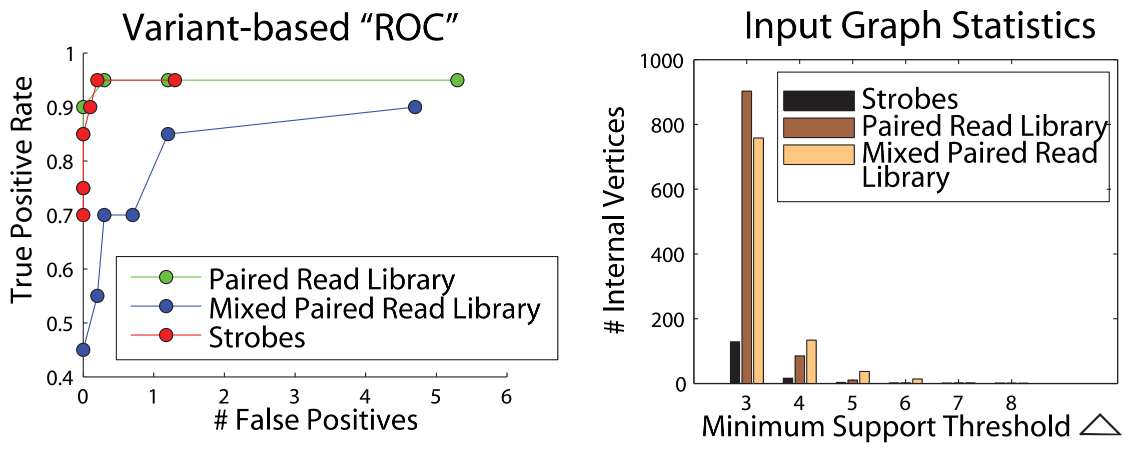

Figure 3.8: **Statistics for ten 10x coverage simulations of a highly repetitive region.** (Left) ROC curve of average number of false positives and average true positive rate, varying Δ from 3 to 8. (Right) Distribution of the number of internal vertices in the generated graph.

data assumes that we are trying to find a small number of SVs, and it has been used successfully by [52] in the context of paired-end reads. However, the generalization from two to t linked reads required a new approach to solve the t-strobe minimum breakpoints problem. We note that for 2-strobes, the t-strobe minimum breakpoints problem is analogous to the problem posed by [52] for paired reads.

We have shown that strobes have advantages in sensitivity and specificity over paired reads for structural variant detection in simulation. In particular, strobe reads nearly double the specificity at fixed sensitivity for structural variation prediction. Our method is applicable to many types of SVs, including deletions, small insertions, inversions, and translocations. Since a single strobe can resolve multiple breakpoints, inference of duplications and rearrangements become more direct. For example, both breakpoints that denote an inversion may be detected by the *same* t-strobe (Figure 3.2).

We can successfully detect SVs in highly-repetitive regions, which is highlighted in the Repeat Dataset where the test genome is comprised of piecing together repetitive regions. Inversions in particular are often difficult to detect because they are sometimes flanked by nearly-identical regions called segmental duplications - in the next chapter we will see examples of these difficult inversions in human genomes.

There are some limitations to the parsimony-based model, however. First, since no strobe sequencing data was available at the time of publication, our simulations were based on a lower sequencing error than was eventually reported. All simulations in this chapter were conducted by adding 5% sequencing error into the reads based on Pacific Biosciences' in-house simulator; however real sequencing error rates are currently reported to be closer to 15%. In the subsequent chapters we design our simulations to reflect the true sequencing error rate. Additionally, the cost of Pacific Biosciences sequencing is currently higher than a typical run of Illumina sequencing at the same sequencing coverage; thus it is currently not feasible to sequence an entire human genome at 30X sequence coverage using PacBio. Future simulations also reflect this by simulating much lower PacBio sequencing coverage.

Additionally, while the simulations clearly show the improvement of multiply-linked subreads over paired reads, the parsimony-based method infers a surprisingly large number of incorrect SVs. Consider the Mixed Paired Read Library and the Strobe simulation in Figure 3.5, which assesses the detectability of 124 deletions. While strobes reduces the number of false positive SVs by over 100 across all sequence coverages, strobes *still* infers between 85 and 120 false positives. This is about the same number as true positives in the set, implying that we have a false discovery rate of about 50%. In the next chapter, we take a closer look at how to further improve the specificity of the method to get to a reasonable false discovery rate.

Chapter 4

SV Detection using Strobe Sequencing: A Probabilistic Approach

In the previous chapter, we introduced a parsimony-based algorithm for predicting SVs from strobe sequencing data [124]. This work was the first method for SV prediction using strobes, but while it showed that there is extra information in multiply-linked subreads compared to paired reads, the total number of false positives inferred on benchmark datasets were about the same as the total number of true positives: thus, there is still room for significant improvement for improving prediction specificity. Further, [124] was not tested on real sequencing data, and the simulations were conducted with an assumed sequencing error rate that was lower than the currently-reported rates.

In this chapter, we describe a probabilistic model to detect SVs from strobe sequencing data and a Markov Chain Monte Carlo (MCMC) method to sample from the space of possible solutions. Additionally, we present a novel representation of the space of potential inferred SVs that provides a more accurate set of coordinates for each SV. The improved performance on benchmark datasets over the parsimony-based method shows that the extra information provided by the probabilistic framework is useful for predicting SVs. Additionally, we show a direct comparison to simulated strobe sequencing and simulated paired-end (e.g. Illumina) sequencing, and demonstrate that combining datasets is even more powerful than the strobe or paired-end dataset alone. Finally, we apply our method to real strobe sequencing data that contain reported SVs from the human genome, and successfully recover all the reported variants.

4.1 Preliminaries

We continue the notation from the Preliminaries section in the previous chapter (§3.2). For a set $\mathbf{S}$ of strobes, let $\mathcal{P}$ be the set of all possible consecutive subread pairs:

$$\mathcal{P} = \bigcup_{S \in \mathbf{S}} \bigcup_{i=1}^{t-1} \left\{ A\left(R_i^{(S)}\right) \times A\left(R_{i+1}^{(S)}\right) \right\}. \tag{4.1}$$

A consecutive subread pair $(a_1, a_2) \in \mathcal{P}$ is *concordant* if the aligned distance and orientation of the pair is expected given the generative model; that is, if the alignment orientations are the same and the aligned distance is within some bounds u, l on the expected advance length. If (a_1, a_2) is not concordant, then it is *discordant* and implies an SV in the test genome. Let $\mathcal{D} \subseteq \mathcal{P}$ be the set of discordant consecutive subread pairs (which we will call discordant pairs when the context is clear). The discordant pairs may be further classified as implying deletions, inversions, and translocations according to the alignment orientation and aligned distance.

Each strobe $S \in \mathbf{S}$ has a set of *strobe alignments* $A(S)$, defined as

$$A(S) = \{A(R_1) \times A(R_2) \times \ldots \times A(R_t)\} \cup \emptyset. \tag{4.2}$$

The empty set $\emptyset$ is an element of $A(S)$ because the correct strobe alignment might be missing from the set of alignments for S. A selection of one strobe alignment from each strobe is called a *mapping* M:

$$M \in (a_s \in A(S) \; \forall \; S \in \mathbf{S}), \tag{4.3}$$

where some elements in M may be duplicates (i.e. in the case where multiple strobes have $\emptyset$ selected).

4.2 A Probabilistic Method for SV Prediction with Strobes

We will first describe the novel representation of the space of potential inferred SVs, then describe the probabilistic model and the MCMC method and present results.

4.2.1 A New Representation of Inferred SVs

Clustering the set of discordant pairs $\mathcal{D}$ determines the possible SVs implied by $\mathbf{S}$. As before, we cluster discordant pairs using GASV [133], which uses a geometric interpretation of discordant pairs as a trapezoid in $\mathbb{R}^2$. Here we briefly describe the geometric interpretation developed by [133], and then we will extend the geometric interpretation of clusters to more accurately report the coordinates implied by subsets of discordant pairs in $\mathcal{D}$.

Geometric Interpretation of Discordant Pairs For a discordant pair $(a_1, a_2) \in \mathcal{D}$, the trapezoid in $\mathbb{R}^2$ is defined as the coordinates (x, y) that could denote the SV breakpoint implied by the pair. If (a_1, a_2) implies a deletion, for example, the points (x, y) are defined by the inequality in Equation 3.1. Similar equations exist for other SV types such as inversions and translocations (see §3.2).

If two discordant pairs $(a_1, a_2), (b_1, b_2) \in \mathcal{D}$ intersect in $\mathbb{R}^2$, the intersection contains possible SV breakpoints (x, y) that are consistent with both discordant pairs. By clustering the discordant pairs $\mathcal{D}$ based on this geometric interpretation, we determine sets of discordant pairs that mutually intersect. The area

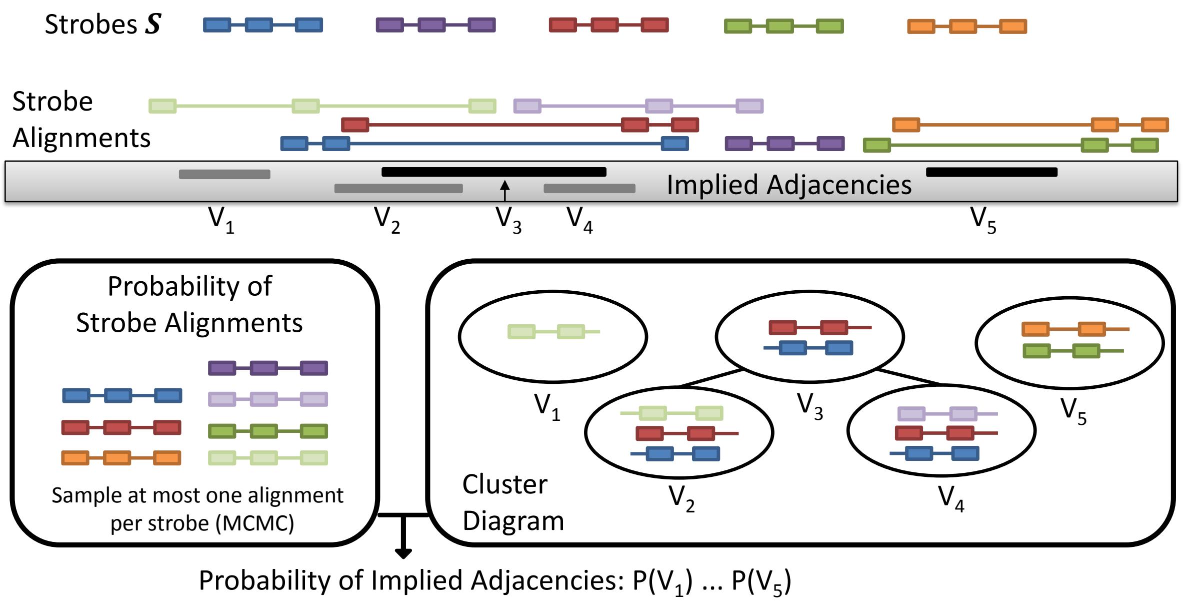
Strobes *S*
Strobe Alignments
Implied Adjacencies
V1
V2
V3
V4
V5
Probability of Strobe Alignments
Sample at most one alignment per strobe (MCMC)
Cluster Diagram
Probability of Implied Adjacencies: P(V1) ... P(V5)

Figure 4.1: **Overview of the probabilistic model for SV prediction from strobe sequencing data.** In this example there are five strobes; three have a unique strobe alignment to the reference (blue, red, and orange), and two have two strobe alignments to the reference (green and purple). There are five adjacencies that are implied by different selections of strobe alignments; suppose that two of them (V_3 and V_5) are correct. The implied adjacencies are represented as a cluster diagram G, where the nodes are the adjacencies and the edges denote interval overlaps. Using an MCMC method, we compute the probability of each strobe alignment in the space by sampling at most one alignment for each strobe with probability proportional to the posterior of the mappings. Finally, these probabilities are used to assign a probability of each implied adjacency (i.e. each node in G).

of intersection is called the *breakpoint region* because the intersection contains all points (x, y) that are consistent with all pairs.

Consider a set of discordant pairs that all pairwise intersect in $\mathbb{R}^2$; often, all discordant pairs have a common area of intersection (Figure 4.2 Top Left). In this case, all discordant pairs imply, or *support*, the same candidate SV. However, when incorrect alignments are present, there may not be a single area of intersection for all discordant pairs (Figure 4.2 Bottom Left). This may be the case in highly-repetitive regions, as well as incorrect alignments due to the high sequencing error rate. In this case, we must identify the collection of possible SVs implied by the overlaps, noting that the breakpoint coordinates (x, y) change depending on the subset of discordant pairs. A previous publication [133] described the concept of *maximal clusters* of discordant pairs in these complicated cases, which are maximal sets of discordant pairs that mutually have an area of intersection. However, the maximal set of discordant pairs may not accurately reflect the true SV coordinate, particularly if there are a small number of incorrect discordant pairs that align to the same location. In the bottom left panel of Figure 4.2, there are two maximal clusters: one contains discordant pairs $\{1, 2, 3, 4\}$ and the other contains discordant pairs $\{2, 3, 4, 5\}$. However, neither of the breakpoint regions overlaps the correct coordinate (x, y). In this sense, either of the maximal clusters are "approximate" in their calling of the SV - they both contain discordant pairs that correctly span the true deletion, but the coordinate implied by the clusters do not contain the real breakpoint.

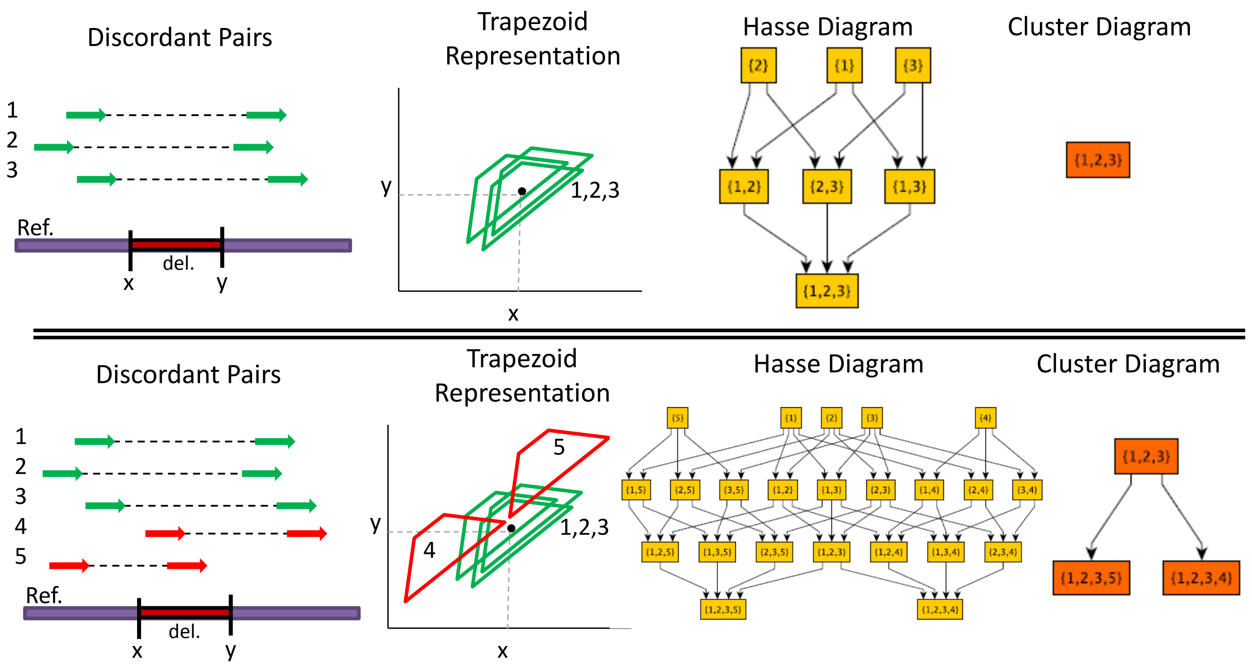

Figure 4.2: **Cluster Diagram Construction.** Cluster Diagrams for (Top) discordant pairs with a common area of intersection and (Bottom) discordant pairs with no common area of intersection. Each discordant pair can be represented as a trapezoid in $\mathbb{R}^2$, and the set of all transitively-reduced intersections is a Hasse diagram (a partially ordered set on the pairs with the subset operator). After successively contracting nodes with a single outgoing edge and removing transitive edges, we get a cluster diagram. For sets of pairs with a common intersection, the cluster diagram is simply a single node.

Comprehensively Describing All Possible Sets of Intersecting Discordant Pairs To gain better accuracy when calling breakpoint regions for inferred SVs, we have developed a novel representation of SVs that describes the entire set of candidate SVs and their breakpoint coordinates. Keeping with the GASV interpretation of discordant pairs as trapezoids in $\mathbb{R}^2$, we construct a graph G, called a *cluster diagram*. Nodes in G represent candidate SVs (sets of discordant pairs that mutually intersect), and directed edges in G describes the relationship between the overlaps. The maximal clusters described by [133] are recovered as a subset of the nodes in the cluster diagram G.

Let H be all sets of discordant pairs in $\mathcal{D}$ that have non-empty intersections; there are as many as $2^{|\mathcal{D}|}$ of them. We first construct the Hasse diagram for the partially ordered set $(H, \subset)$, which we represent as a directed acyclic graph. Nodes v with a single outgoing edge to u indicate that the set of discordant pairs in v mutually intersect with *only* the set in u; thus we contract all nodes with a single outgoing edge. We call the graph $G = (V, E)$ a *cluster diagram*, and is built in the following steps:

1. Construct $G = (V, E)$, where the nodes $V = H$ are sets of discordant pairs with a non-empty intersection, and there is a directed edge (u, v) if $u \subset v$.
2. Remove all edges implied by the transitive property. That is, if $u \subset v$ and $v \subset z$, remove the edge (u, z).
3. Contract all nodes $v \in V$ with a single outgoing edge.
4. Repeat steps 2 and 3 until all nodes have outdegree of at least 2.

There are two types of nodes in G: the leaves of G are the maximal sets of discordant pairs with a non-empty intersection, and the other nodes in G are sets of discordant pairs that appear in multiple maximal sets. If all discordant pairs have a common area of intersection, then it will be represented as a single node in the cluster diagram G (Figure 4.2). Note that the same discordant pair may appear in multiple nodes in G, contributing to different sets of possible SV breakpoint coordinates.

An important step for constructing the cluster diagram is determining H, the sets of discordant pairs in $\mathcal{D}$ that have non-empty intersections. For $|\mathcal{D}|$ discordant pairs, there can be as many as $2^{|\mathcal{D}|}$ possible sets in H. Instead of computing the intersections for all $2^{|\mathcal{D}|}$ sets, we utilize the fact that these are represented as convex polygons in $\mathbb{R}^2$. We use the following theorem from computational geometry:

Helly's Theorem [154] Suppose we have $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ convex sets in $\mathbb{R}^d$, where $n > d$. If the intersection of all combinations of $d+1$ convex sets is nonempty, then the intersection of all n convex sets is nonempty. Mathematically, let $\mathbb{Y}$ be all $\binom{n}{d+1}$ sets of convex sets, where an element $\mathbf{Y} \in \mathbb{Y}$ is a set of convex sets, that is $\mathbf{Y} \subseteq \mathbf{X}$. If

$$\bigcap_{X_i \in \mathbf{Y}} X_i \neq \emptyset \;\; \forall \;\; \mathbf{Y} \in \mathbb{Y}, \tag{4.4}$$

then

$$\bigcap_{i=1}^{n} X_i \neq \emptyset. \tag{4.5}$$

The trapezoids in $\mathbb{R}^2$ are convex, so we only compute the intersections for $\binom{|\mathcal{D}|}{3}$ sets of discordant pairs. From these sets, we can determine the entire set H.

4.2.2 Probabilistic Model

Now that we have a graph that represents all possible SVs given the strobe alignments from $\mathbf{S}$, we turn our attention to the probabilistic model. For each SV (and hence each node in the cluster diagram G), we wish to compute the marginal probability of the SV given the data. However, the probability of observing the SVs from the data depend on the strobe alignments selected for each strobe. So, we first compute the joint probability of observing the strobe alignments as mappings M. The rest of this section proceeds as follows:

1. We first derive the probability of the mappings M given the data. Here, the data is the set of all possible strobe alignments $A(S)$ for each strobe $S \in \mathbf{S}$, which in turn produces the implied SVs (represented as a cluster diagram G).
2. We then describe the Markov Chain Monte Carlo (MCMC) algorithm to compute the posterior probability of the mappings.
3. Finally, we describe how to compute the marginal probability of the SVs (i.e. the nodes in G) using the joint probability of the alignments.

The first goal is to compute the probability of a mapping M given the data. Let $A(\mathbf{S})$ be all possible strobe alignment sets.

$$P(M|A(\mathbf{S}), G) = \frac{P(M)P(A(\mathbf{S}), G|M)}{P(A(\mathbf{S}), G)} \tag{4.6}$$

$$\propto P(M) \times P(A(\mathbf{S})|M) \times P(G|M), \tag{4.7}$$

where $A(\mathbf{S})$ is the set of strobe alignments for all strobes $S \in \mathbf{S}$. There are three main terms: (1) the prior $P(M)$, (2) the probability of the selected subread alignments $P(A(\mathbf{S})|m)$, and the probability of the implied SVs $P(G|M))$. We assume a uniform prior for $P(M)$, meaning that the mappings are equally likely. The other two terms are discussed in the following sections.

Computing $P(A(\mathbf{S})|M)$. For a mapping M, there let $A(M)$ be the set of subread alignments in the mapping, and let e_M be the number of strobes for which the empty set $\emptyset$ was selected. The probability of subread alignments conditioned on M is independent of all subread alignments not in M: Thus, $P(A(\mathbf{S})|M) = P(A(M))$. The probability $P(A(M))$ of the selected subread alignments depends on the sequencing technology; thus we have a probability p_{seq} of the sequencing error (this may be set in advance or inferred from the alignments). For a mapping M, let $\epsilon(M) = \sum_{a \in A(M)} \epsilon_a$ be the total number of errors and $\ell(M) = \sum_{a \in A(M)} (y_a - x_a + 1)$ be the total aligned length of all subread alignments in M. We model the probability of observing $\epsilon(M)$ errors in a string of length $\ell(M)$ when the sequencing error is p_{seq} as

$$Bin\left(\epsilon(M); \ell(M), p_{seq}\right) = \binom{\ell(M)}{\epsilon(M)} p_{seq}^{\epsilon(M)} (1 - p_{seq})^{\ell(M) - \epsilon(M)}.$$

We assume a fixed probability p_{err} that the correct strobe mapping is not present in the collection of alignments. We approximate the Binomial term with a Normal distribution when $\ell(M) p_{seq} > 6$ (which is often the case).

Computing $P(G|M)$. To compute $P(G|M)$, we determine the implied SVs from the discordant pairs in M and compute the probability of observing those SVs. Once a mapping M is selected, this determines a subset $\mathcal{D}(M) \subseteq \mathcal{D}$ of discordant pairs. We need to determine the SVs implied by $\mathcal{D}(M)$, assuming that each discordant pair implies exactly one SV. To do so, we find the smallest number of nodes in the cluster diagram G that cover the discordant pairs in $\mathcal{D}(M)$, which can be efficiently computed using a set cover approximation on the leaves of the diagram. Let $L \subseteq V$ be the set of leaves in G; remember that each leaf node is a set of discordant pairs from $\mathcal{D}$. Given a mapping M, we have a subset $\mathcal{D}(M)$ of discordant pairs. The algorithm SetCover($D \subseteq \mathcal{D}$,$U \subseteq V$) takes a set of discordant pairs and a set of nodes and returns the smallest set of nodes that cover all the discordant pairs in D (Algorithm 1) *and* the discordant pairs from D that are covered by each node. Thus, SetCover(D,U) returns sets of discordant pair sets, and each discordant pair set is a subset of some node in U.

Algorithm 1 SetCover(D,U)

```
1: $\tilde{U} \Leftarrow \emptyset$
2: **while** $D \neq \emptyset$ **do**
3:     $u' \Leftarrow \arg\max_{u \in U} u \cap D$ // Get node containing the most pairs in the current set.
4:     $d' \Leftarrow u' \cap D$ // set of discordant pairs in $u'$
5:     $\tilde{U} \Leftarrow \tilde{U} \cup \{d'\}$ // Add the set of pairs
6:     $D \Leftarrow D \setminus d'$ // Remove pairs from consideration.
7: **end while**
8: **return** $\tilde{U}$
```

Running SetCover($\mathcal{D}(M)$,L) produces a set $\tilde{U}$ of discordant pair sets, where each discordant pair sets represents an implied SV. The *support* of the implied SV is the number of discordant pairs in the associated set. Let the vector of supports be $\Phi(M)$, which has length $|U|$.

We assume that SVs appear independently in the test genome, so each coordinate in the test genome is expected to be supported by λ strobes. Since each strobe contains $t - 1$ consecutive subread pairs, each coordinate is supported by

$$\lambda_d = L_{avg}\lambda(t-1) \tag{4.8}$$

consecutive subread pairs, where L_{avg} is the average advance length. Thus, we model the expected support of an SV as a Poisson process with parameter λ_d. The probability of SVs with supports $\Phi(m)$ is then

$$\begin{aligned} P(G|M) = P(\Phi(M)) &= \prod_{k\in\Phi(M)} P(\text{SV has support } k) && (4.9)\\ &= \prod_{k\in\Phi(M)} \text{Poiss}(k;\lambda_d) = \prod_{k\in\Phi(M)} \frac{e^{-\lambda_d}\lambda_d^k}{k!}. && (4.10)\end{aligned}$$

Thus, the full model is

$$P(M|A(\mathbf{S}),G) \propto P(M)Bin\left(\epsilon(M);\ell(M),p_{seq}\right) \times p_{err}^{e_M} \times \prod_{k\in\Phi(M)} \text{Poiss}(k;\lambda_d) \tag{4.11}$$

with hyperparamters p_{seq},p_err, and λ_d.

A Note on the Prior $P(M)$**.** For the results presented below, we assume a uniform prior $P(M)$, which drops out of Equation 4.11. Other methods such as [134] employ an exponential prior on the number of implied SVs, utilizing the parsimony assumption that we seek a relatively few number of SVs. We also used this prior for a number of early simulations, but we ultimately found that the exponential prior was "too strong" because the Poisson probability of the SV supports combined with the probability of the alignment qualities naturally limited the number of SVs inferred by the method; that is, the exponential prior drove many of the SV probabilities to zero.

Finally, we note that while we use a uniform prior $P(M)$, this is not necessarily a uniform prior on the strobe alignments. For example, suppose that there one strobe has a single, unique strobe alignment and another strobe has a number of ambiguous strobe alignments. If we consider the space of mappings, a flat prior $P(M)$ will provide a much larger prior strobe alignment probability for the unique strobe alignment compared to the ambiguous strobe alignments.

4.2.3 Generalizing to Multiple Data Types

Suppose, rather than one set of strobes $\mathbf{S}$, we have E sets of strobes $\{\mathbf{S}^{(1)},\ldots\mathbf{S}^{(E)}\}$ corresponding to E different experiments. $\mathbf{S}^{(e)}$ may also be paired-read data from next generation sequencing technologies, which are analogous to 2-strobes (see §3.2). For each experiment $1 \leq e \leq E$, we have the following information:

1. $M(\mathbf{S}^{(e)})$: the set of strobe alignments

2. $\lambda_d^{(e)}$: the expected number of strobes that supports an SV
3. $p_{seq}^{(e)}$: the sequencing error rate
4. $p_{err}^{(e)}$: the probability that a strobe is not aligned

We cluster the union of the strobe alignments $\{A(\mathbf{S}^{(1)}),\ldots,A(\mathbf{S}^{(E)})\}$ and get a cluster diagram $G = (V, E)$. For a node $v \in V$, let $v^{(e)}$ be the set of pairs that belong to experiment e. Finally, for a mapping vector M let $M^{(e)}$ be the set of indices that correspond to strobes from experiment e. Then,

$$\begin{aligned}
P(M|A(\mathbf{S}^{(1)}),\ldots,A(\mathbf{S}^{(E)}),G) &\propto P(M) \times P(A(\mathbf{S}^{(1)}),\ldots,A(\mathbf{S}^{(E)})|M) \times P(G|M) && (4.12)\\
&= \prod_{e=1}^{E}\left[P(M^{(e)}) \times P(A(\mathbf{S}^{(e)})|M^{(e)}) \times P(G|M^{(e)})\right] && (4.13)\\
&= \prod_{e=1}^{E}\left[Bin\left(\epsilon(M^{(e)});\ell(M^{(e)}),p_{seq}^{(e)}\right) \times p_{err}^{e_M^{(e)}} \times \prod_{k\in\Phi(M^{(e)})}\text{Poiss}(k;\lambda_d^{(e)})\right] && (4.14)
\end{aligned}$$

Note that we could weigh the different experiments based on external information - in this dissertation, all experiments are weighed equally.

4.2.4 Markov Chain Monte Carlo (MCMC)

The probability in 4.11 is computationally prohibitive because the number of possible strobe mappings M is large. Thus, we develop a Markov Chain Monte Carlo (MCMC) method to sample mappings M with probability proportional to $P(M|A(\mathbf{S}),G)$ using the Metropolis-Hastings algorithm.

The MCMC Algorithm Algorithm 2 takes the set of alignments $A(\mathbf{S})$, a cluster diagram G, and a number of iterations z, and returns a set of $z+1$ mappings M that are sampled with probability proportional to $P(M|A(\mathbf{S}),G)$.
This is an implementation of a lazy chain, where the mapping M stays the same with probability $\frac{1}{2}$. There are two ways this chain moves through the solution space: via local moves that change the assignment of a single strobe and via jump moves that change the assignment of multiple strobes. β is a user-defined parameter that determines the proportion of local vs. jump moves: we set $\beta = 0.9$.

Let $A'(S)$ be the set of strobe alignments *excluding* the empty set $\emptyset$ (this is useful for presenting the algorithms). Algorithm 3 describes the local move, which takes the set of strobe alignments and a mapping vector and returns the vector with a single entry changed.

Algorithm 4 describes the jump move, which moves sets of strobe alignments. To ensure that the move is symmetric and not a naive move, we perform this move on a subset of connected components $\tilde{G} = \{G_1, G_2, \ldots, G_k\}$ of G. Consider a connected component $G_k \subseteq G$ of the cluster diagram; there is a set

Algorithm 2 MCMC($A(\mathbf{S})$,G,z)

```
1: Initialize $M^{(0)}$ with a random assignment of mappings
2: for $i = 1 \to z$ do
3:     $M' \Leftarrow \begin{cases} M^{(i-1)} & \text{with probability } \frac{1}{2} \\ \text{makeLocalMove}(M^{(i-1)}, M(\mathbf{S})) & \text{with probability } \frac{\beta}{2} \\ \text{makeJumpMove}(M^{(i-1)}, M(\mathbf{S})) & \text{with probability } \frac{1-\beta}{2} \end{cases}$
4:     $ratio \Leftarrow \dfrac{P(M'|A(\mathbf{S}), G)q(M^{(i-1)}|M')}{P(M^{(i-1)}|A(\mathbf{S}), G)q(M'|M^{(i-1)})}$
5:     $M^{(i)} \Leftarrow \begin{cases} M' & \text{with probability } \alpha(M', M^{(i-1)}) = \min(1, ratio) \\ M^{(i-1)} & \text{otherwise} \end{cases}$
6: end for
7: return $\{M^{(0)}, M^{(1)}, M^{(2)}, \ldots, M^{(z-1)}, M^{(z)}\}$
```

Algorithm 3 makeLocalMove($A(\mathbf{S})$,M)

```
1: Select strobe $S$ uniformly from $\mathbf{S}$.
2: $a \Leftarrow$ the current alignment in mapping $M$ from $A(S)$ (which may be $\emptyset$)
3: $a_{old} \Leftarrow a$
4: while $a = a_{old}$ do
5:     $a \Leftarrow \begin{cases} \text{Select } \emptyset & \text{with probability } p_{err} \\ \text{Otherwise, select } a \in A'(S) & \text{uniformly from } A'(S) \end{cases}$
6: end while
7: return $M$
```

of strobes $S \subseteq \mathbf{S}$ that have discordant pairs present in G_k. G_k is in $\tilde{G}$ if there are at least two strobes $S_i, S_j \in S$ such that $|A'(S_i)| = |A'(S_j)| = 1$. Since there is only one alignment for each of these strobes, then a move is deterministic (it moves from an error to the alignment or vice versa). Since there are at least two alignments with this characteristic, then moving the alignments of all such strobes cannot be done with a local move.

The Proposal Distribution

Observe that for any two mapping vectors M and M', there can only be *one* type of move that is feasible: no move, the local move, or the jump move. Thus, the proposal distribution is described as follows:

$$q(A'|A) = \begin{cases} 1 & \text{if } M = M' \text{ (the lazy chain).} \\ q_{local}(M'|M) & \text{if we can move from } M \text{ to } M' \text{ using a local move.} \\ q_{jump}(M'|M) & \text{if we can move from } M \text{ to } M' \text{ using a jump move.} \end{cases} \tag{4.15}$$

Below we describe the calculations required for q_{local} and q_{jump}.

Algorithm 4 makeJumpMove($A(\mathbf{S})$,M)

```
1: Select connected component $G_k$ uniformly from $\tilde{G}$.
2: for $S \in V_k$ do
3:     if $|A'(S)| = 1$ then
4:         $a \Leftarrow$ the current alignment in mapping $M$ from $A(S)$ (which may be $\emptyset$ or the single alignment)
5:         if $a = \emptyset$ then
6:             $a \Leftarrow A'(S)$
7:         else
8:             $a \Leftarrow \emptyset$
9:         end if
10:    end if
11: end for
12: return $M$
```

q_{local}**:** Suppose that we have made a move by selecting strobe S with probability $1/n$. Thus, we have proposed a mapping vector M' after calling makeLocalMove($A(\mathbf{S})$,M). Note that M and M' differ only by the alignment for strobe S, which has $|A'(S)|$ possible strobe alignments (excluding the empty set). Let a be the strobe alignment in M and a' be the strobe alignment in M' (by definition, $a \neq a'$). The probability of proposing M' from M is

$$q_{local}(M'|M) = \frac{1}{n} \times \begin{cases} 1 & \text{if } a \neq \emptyset \text{ and } a' = \emptyset \text{ and } |A'(S)| = 1 \\ p_{err} & \text{if } a \neq \emptyset \text{ and } a' = \emptyset \text{ and } |A'(S)| > 1 \\ \frac{1}{|A'(S)|} & \text{if } a = \emptyset \text{ and } a' \neq \emptyset \\ \frac{1-p_{err}}{|A'(S)|-1} & \text{if } a \neq \emptyset \text{ and } a' \neq \emptyset \end{cases}.$$

Conversely, the probability of proposing M from M' is

$$q_{local}(M|M') = \frac{1}{n} \times \begin{cases} 1 & \text{if } a = \emptyset \text{ and } a' \neq \emptyset \text{ and } |A'(S)| = 1 \\ p_{err} & \text{if } a = \emptyset \text{ and } a' \neq \emptyset \text{ and } |A'(S)| > 1 \\ \frac{1}{|A'(S)|} & \text{if } a \neq \emptyset \text{ and } a' = \emptyset \\ \frac{1-p_{err}}{|A'(S)|-1} & \text{if } a \neq \emptyset \text{ and } a' \neq \emptyset \end{cases}.$$

q_{jump}**:** Suppose that we have made a move by selecting connected component G_k with probability $1/|\tilde{G}|$. Thus, we have proposed a mapping vector M' after calling makeJumpMove($A(\mathbf{S})$,M). By definition, all the strobes with unique alignments that support G_k are flipped (if they are errors in M they are set to the alignment in M', and if they are alignments in M they are set to errors in M'). Thus,

$$q_{jump}(M|M') = q_{jump}(M'|M) = \frac{1}{|\tilde{G}|}.$$

Independent Subproblems and Sampling from the Chain Running MCMC($A(\mathbf{S})$,G,z) on the entire space of strobes $\mathbf{S}$ would require a very long convergence time, since the number of possible solutions grows exponentially with the number of strobes. However, the repetitive nature of the reference genome allows us to divide $\mathbf{S}$ into independent subsets for which the MCMC algorithm can be run in parallel. Two strobes $S_1, S_2 \in \mathbf{S}$ are *dependent* if they have discordant pairs that appear in the same node in the cluster diagram G; from this definition, we determine sets of strobes that are mutually dependent and independent of all other strobes. We run the MCMC chain for two-to-ten million iterations, depending on the number of strobes and alignments in the subproblem.

Rather than recording all mapping vectors M sampled by the chain, in practice we run the chain for a number of iterations before we start recording the sampled states. In all experiments we had a burnin time of 90% of the iterations, and recorded the last 10%. We also tested our method with a shorter burnin (10% burnin rate) and a thinning method (sampling every 100,000th iteration); however, the longer burnin of 90% produced the most stable results due to the convergence of the chain.

4.2.5 Predicting Structural Variants from the MCMC Method

The MCMC method returns a set of $z+1$ mappings $\{M^{(0)}, M^{(1)}, M^{(2)}, \ldots, M^{(z-1)}, M^{(z)}\}$; however, ultimately we aim to predict SVs (which are represented as nodes in the cluster diagram G). One option is to select a single mapping $\hat{M}$ from the chain (e.g. the M with the highest probability, M averaged over all possible samples, etc.) and determine the nodes in G that cover the discordant pairs $\mathcal{D}(\hat{M})$. However, we lose information when selecting a single mapping M from the chain; instead, we have designed a metric that utilizes the probabilities for all sampled mappings. We describe how to compute an *SV probability* for each node in the cluster diagram G. This probability, computed for each node v in the cluster diagram G, is the marginal probability of the node; however, we use the joint probability of the strobe alignments to compute these marginal probabilities.

Consider a node v in the cluster diagram with discordant pairs $v = \{d_1, d_2, \ldots d_{|v|}\} \subseteq \mathcal{D}$. If v is a correct SV, then we assume that at least some of the discordant pairs in v are correct. Further, we assume that other discordant pairs *not* in v that intersect at least one of $\{d_1, d_2, \ldots d_{n_v}\}$ must be incorrect. Call these pairs $\bar{v} = \{\bar{d}_1, \bar{d}_2, \ldots, \bar{d}_{|\bar{v}|}\}$. As a concrete example, consider the right child in the cluster diagram in Figure 4.2 bottom. Here, there are four discordant pairs in $v : \{d_1, d_2, d_3, d_4\}$, and one discordant pair in $\bar{v} : \{d_5\}$.

We wish to compute the probability that v is supported by k discordant pairs for $k = 0, 1, \ldots, |v|$. First, however, we must determine the probability of a single discordant pair $d \in \mathcal{D}$. The discordant pair d may appear in a number of mappings M; so we simply count the number of times the discordant pair appears in the chain of $\{M^{(0)}, M^{(1)}, M^{(2)}, \ldots, M^{(z-1)}, M^{(z)}\}$:

$$P(d) = \frac{1}{z+1} \sum_{i=0}^{z} \mathbf{1}_{d \in \mathcal{D}(M^{(i)})}. \tag{4.16}$$

To compute the probability that v is supported by k discordant pairs, we could compute all $\binom{|v|}{k}$ options. In our example above, for $k = 3$ we compute the following:

$$P(v = \{d_1, d_2, d_3, d_4\}, \bar{v} = \{d_5\} \text{ is supported by 3 discordant pairs}) = \tag{4.17}$$

$$[P(d_1) \times P(d_2) \times P(d_3) \times (1 - P(d_4)) \times (1 - P(d_5))] + \tag{4.18}$$

$$[P(d_1) \times P(d_2) \times (1 - P(d_3)) \times P(d_4) \times (1 - P(d_5))] + \tag{4.19}$$

$$[(1 - P(d_1)) \times P(d_2) \times P(d_3) \times P(d_4) \times (1 - P(d_5))]\,. \tag{4.20}$$

We assume the discordant pairs are independent, so each term is simply a product of discordant pair probabilities determined by Equation 4.16. For nodes with many discordant pairs in v, however, this enumerative method is prohibitively slow. Instead, we have a dynamic program takes as input v and $\bar{v}$ and returns the probability that v is supported by k discordant pairs for $k = 0, 1, \ldots, |v|$ (Algorithm 5). The method fills a 0-indexed table T, where T_{ij} is the probability that the node is supported by i discordant pairs out of the first j pairs (which are arbitrarily ordered). Lines 2-6 in NodeProbability(v,$\bar{v}$) are initializations for the recurrence. There are two ways that the node can be supported by i discordant pairs in the first j pairs in the recursive step (Line 9): either the jth pair is included in the count (and we use the probability in $T_{(i-1)(j-1)}$ or it is not (and we use the probability in $T_{(i-1)j}$). Lines 12-14 account for the discordant pairs in $\bar{v}$ by multiplying T by the probability that these are not selected (as these would denote other nodes in the cluster diagram G).

Algorithm 5 NodeProbability(v,$\bar{v}$)

```
1: $T$ is a $(|v|+1)$-by-$(|v|+1)$ table
2: $T_{00} \Leftarrow 1$
3: $T_{ij} \Leftarrow 0$ for $i > j$
4: **for** $j \Leftarrow 1$ to $|v|$ **do**
5:     $T_{0j} \Leftarrow T_{0(j-1)} \times (1 - P(d_j))$
6: **end for**
7: **for** $i \Leftarrow 1$ to $|v|$ **do**
8:     **for** $j \Leftarrow i$ to $|v|$ **do**
9:         $T_{ij} = T_{(i-1)(j-1)} \times P(d_j) + T_{i(j-1)} \times (1 - P(d_j))$
10:    **end for**
11: **end for**
12: **for** $i \Leftarrow 0$ to $|v|$ **do**
13:     $T_{i|v|} \Leftarrow T_{i|v|} \times \prod_{\bar{d} \in \bar{v}} (1 - P(\bar{d}))$
14: **end for**
15: **return** $T_{i|v|}$ for $0 \le i \le |v|$
```

The probability that v is supported by k or more discordant pairs is

$$P(v \text{ has } \geq k \text{ discordant pairs}) = \sum_{i=k}^{|v|} T_{i|v|}. \tag{4.21}$$

When there are multiple experiments (see §4.2.3), we have a $k^{(e)}$ for each experiment, and the probability is computed as

$$P(v \text{ has } \geq k^{(e)} \text{ discordant pairs for experiments } 1 \leq e \leq E) = \frac{1}{E}\sum_{i=1}^{E}\sum_{i=k^{(e)}}^{|v^{(e)}|} T_{i|v^{(e)}|}. \tag{4.22}$$

4.3 Results

We formulate the problem of predicting structural variants from a dataset of multiply-linked reads, which we solve with a probabilistic model of observing these structural variants. We benchmark our model on simulated test chromosomes for low-coverage strobe sequencing, high-coverage paired-end sequencing, and a mixture of strobe and paired-end data. Finally, we demonstrate the accuracy of our model by applying our model to real strobe sequencing data from two fosmids containing known deletions and two fosmids containing known inversions and correctly detect the reported variants.

A Model of Structural Variation

We model a test genome generated from a reference genome by independently adding, duplicating, and rearranging segments of the reference genome. Each pair of adjacent coordinates in the test genome that are not adjacent in the reference genome is called a *novel adjacency*. The number of novel adjacencies in the test genome is related to the number of rearranged segments, or structural variants; some variants, such as deletions, create one adjacency whereas other variants, such as inversions, create two adjacencies.

To identify novel adjacencies in a test genome, long reads and strobes generated from the genome are formalized as an ordered set of contiguous substrings (or *reads*) from the test genome, which we will term a *t-multiread* consisting of t reads $\{R_1, R_2, \ldots, R_t\}$. From a set $\mathbf{S}$ of n t-multireads, we wish to infer the novel adjacencies in the test genome.[1] Each read R_i from a t-multiread S has a set of *read alignments* $A(R_i^{(S)})$ to the reference genome, each of which consists of an interval in the reference genome, an orientation, and the edit distance to the reference. An alignment for a t-multiread S corresponds to selecting a read alignment for each of the t reads. Let $A(S)$ be the set of all combinations of such selections, along with the empty set (which indicates that the correct alignment is not present). The correct alignment for t-multiread S is thus an element in $A(S)$.

A selection of one alignment for each t-multiread $S \in \mathbf{S}$ results in a candidate *mapping* M for the set $\mathbf{S}$ of t-multireads. The goal of this work is to solve the following problem:

t-Multiread Mapping Problem: Given a set $\mathbf{S}$ of t-multireads and their read alignments, find the correct mapping M^* (that is, a selection of one element from $A(S)$ for each $S \in \mathbf{S}$) and the set of novel adjacencies

[1]The value of t might be different for t-multireads in $\mathbf{S}$. For long reads, and long strobe subreads, this notation must be generalized to account for overlapping subsequences.

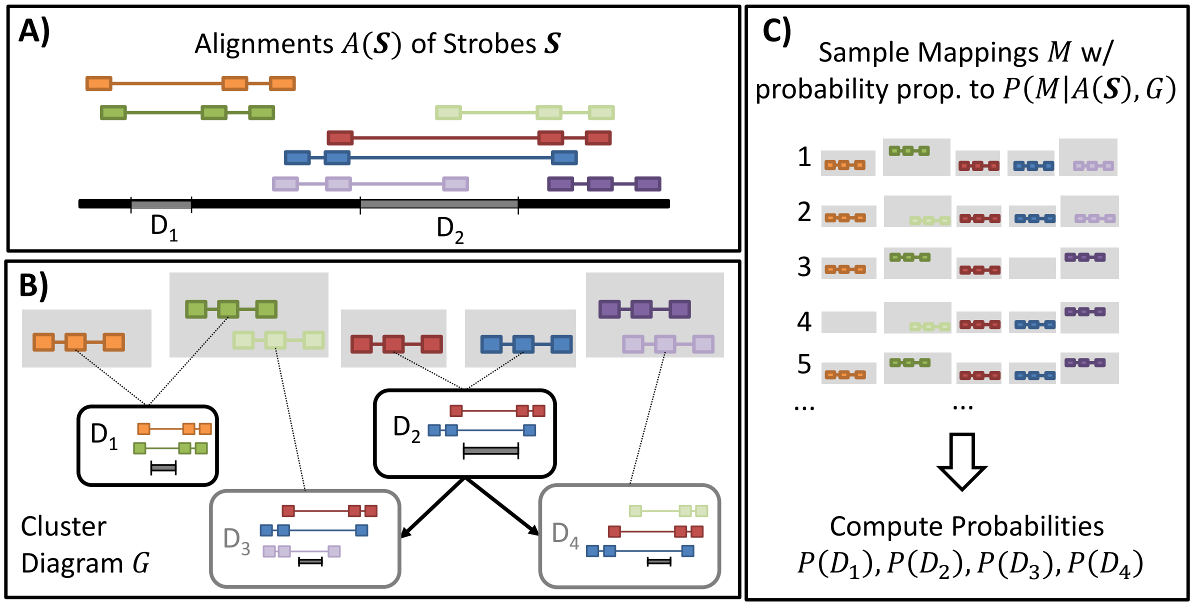
A)
Alignments $A(\boldsymbol{S})$ of Strobes $\boldsymbol{S}$
D_1
D_2
B)
D_1
D_2
D_3
D_4
Cluster Diagram G
C)
Sample Mappings M w/ probability prop. to $P(M|A(\boldsymbol{S}), G)$
1
2
3
4
5
…
…
Compute Probabilities
$P(D_1), P(D_2), P(D_3), P(D_4)$

Figure 4.3: **Overview of the Probabilistic Algorithm for SV Prediction.** In this example, there are five strobes in the set **S**: orange, green, red, blue, and purple. A) There are two strobe alignments for the green and the purple strobes, and the rest of the strobes have a single strobe alignment. The correct strobe alignments (the orange, dark green, red, blue, and dark purple alignments) support two novel adjacencies (D_1 and D_2). However, the light green and light purple alignments introduce two incorrect novel adjacency predictions. B) The space of all possible novel adjacencies is represented as a cluster diagram G, where the nodes are novel adjacencies and the directed edges represent overlapping novel adjacencies. The edges from the strobes to the nodes are not part of G, but are shown here for clarification. C) A solution to the **t-Multiread Mapping Problem** is a selection of at most one alignment for each strobe in **S**, called a *mapping*, which supports some subset of the deletions. We sample mappings from the solution space proportional to the mapping's posterior probability; from these sampled mappings, we finally compute the probabilities of the four deletions.

that M^* implies.

Structural variant detection methods that handle ambiguous data such as [52, 119, 134] will not ensure that a single strobe alignment is chosen for t-multireads when $t > 2$ because the dependence between consecutive read pairs is destroyed (that is, paired-read methods will not ensure that a strobe alignment from $A(S)$ is selected for each S).

A solution to the **t-Multiread Mapping Problem** has two parts: the mappings and the adjacency predictions. In the Methods section we describe a probabilistic model for a mapping M given the data $\mathbf{S}$. This model incorporates two pieces of information from the alignments in M: the quality of the read alignments and the implied novel adjacencies when we cluster pairs of alignments from M. We compute the probability of M using a Markov Chain Monte Carlo (MCMC) method to address the first part of the **t-Multiread Mapping Problem**. To determine the adjacency predictions, we represent the space of possible novel adjacencies as a graph where the nodes are novel adjacencies and the edges connect novel adjacencies whose coordinates overlap. We use this graph, called a *cluster diagram*, in addition to the mapping probabilities to compute the probabilities of each novel adjacency (Figure 4.3).

4.3.1 Simulations

We assessed our method's ability to detect deletions on two simulated test chromosomes: Chromosome 1 with hundreds of novel adjacencies inserted from 1000 Genomes individuals [32] and Chromosome 17 with thousands of novel adjacencies inserted from Craig Venter's genome [73]. We simulated three types of datasets from these test chromosomes:

Strobe Datasets. 3-multireads at 1X, 2X and 5X sequence coverage, with subread and advance lengths distributions determined by the empirical fosmid data.

Paired Datasets. 100bp paired-end reads with 400bp inserts at 30X sequence coverage.

Hybrid Datasets. 1X and 2X 3-multireads combined with 30X paired-reads.

For the strobe datasets, we inserted 15% error using Alchemy [21] and aligned the strobes to the reference chromosome with BLASR [22]. For the paired datasets, we inserted 1% error using wgsim [75] and aligned the reads to the reference chromosome with BWA [74]. Though BWA retains unique alignments, we find that including ambiguous alignments in the paired datasets decreased performance. Each dataset was evaluated for:

Variant Calling Accuracy. Predictions where the discordant pairs imply a novel adjacency within a specified bound of the true, simulated coordinates. This is similar to the double uncertainty metric introduced by [133].

Alignment Assignment Accuracy. Assignments for which there is at least 80% overlap between the true coordinate interval and the interval returned by alignment.

We compare the results produced by our probabilistic method to other methods by plotting the number of true positive and false positive predictions while tuning some sensitivity parameter. For probabilistic methods such as ours and GASVPro [134], we vary the minimum probability of a novel adjacency prediction. Otherwise we vary the minimum Hydra score [119] or the minimum support of a novel adjacency prediction [124, 133]. We normalize assignment accuracy to plot a true receiver operating characteristic (ROC) curve. Variant calling accuracy remains unnormalized since the number of false positive variants is not strictly bounded.

1000 Genomes Simulation

We generated a simulated dataset based on validated variants from the 1000 Genomes Project [32]. There are 734 deletions, 214 mobile element insertions, 26 tandem duplications, and 4 novel sequence insertions reported on chromosome 1 for the individuals sequenced at low coverage and NA12878, which was sequenced to high coverage. We removed overlapping variants and inserted the remaining 794 rearrangements (557 deletions and 237 insertions) in the reference. Of these deletions, 219 (42.86%) have repetitive sequence spanning both of the novel adjacency coordinates.

We assessed the accuracy of our model by comparing to other methods (Figure 4.4). First, we compared to a parsimony method for strobe data which minimizes the number of predicted variants [124]. Our probabilistic method improves over the parsimony method in both specificity and sensitivity in assigning variants and alignments. For example, at a fixed specificity of 8 false positive variants the probabilistic method recovers 82.24% more true positive predictions at 2X coverage and 83.32% at 5X coverage.

The paired dataset outperforms all other datasets in variant calling accuracy, regardless of the variant prediction algorithm. Despite the lack of improvement over paired-end data in terms of the adjacency predictions, the probabilistic method more accurately determines alignments (Figure 4.4) for the paired dataset, improving over Hydra-HQ, GASV, and even GASVPro, which uses an additional signal from read depth. Because ambiguous paired-end data was not passed into the method, this suggests that our method is robust at eliminating spurious/in-correct alignments. The majority of the novel adjacency predictions are true positive variants, each with a small number of incorrect alignments. These incorrect alignments are most often caused by split reads (reads which span a novel adjacency). While the paired dataset improves over the strobe and hybrid datasets in variant calling accuracy, the strobe and hybrid datasets enrich for correct alignments better than the paired dataset, which demonstrates that the probabilistic method can utilize the alignment quality to enrich for correct alignments when multiple alignments are possible. Additionally, this suggests that even with low-accuracy reads the presence of additional subreads and increased read length can lead to increased alignment accuracy.

The strobe datasets return a large number of ambiguous alignments due to their higher sequencing error rates. These ambiguous alignments create groups of t-multiread alignments where distinct subsets yield different, inconsistent, novel adjacency predictions (see Methods). For example, In the 5X strobe

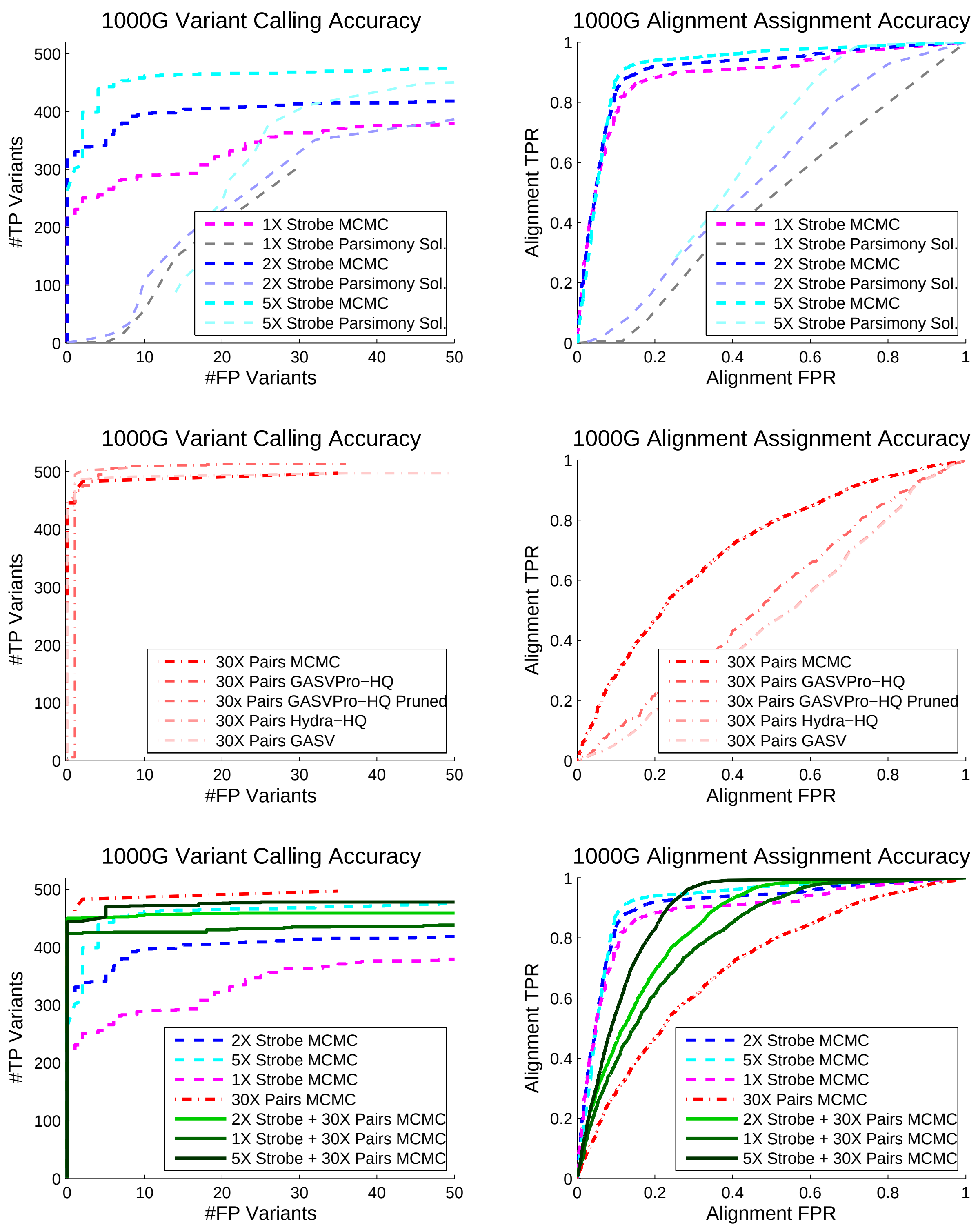

Figure 4.4: **(Left) Variant calling accuracy and (Right) alignment assignment accuracy for datasets from the 1000 Genomes simulation.** Dashed lines are strobe datasets, dot-dash lines are paired datasets, and solid lines are hybrid datasets. The plots are plotted as (top) MCMC vs. Parsimony Solution for 1X,2X, and 5X strobes, (middle) the paired dataset run with a variety of methods, and (bottom) all the datasets run with the probabilistic method. Boldface lines are datasets run with our model: we compared the strobe datasets to a parsimony solution [124] and the paired datasets to Hydra [119], GASV [133], and GASVPro [134]. GASVPro-HQ is the original predictions output by GASVPro on unique alignments, and GASVPro-HQ Pruned removes overlapping variants from the predictions. Hydra-HQ is the Hydra method applied to unique alignments.

dataset, incorrect ambiguous alignments result in inconsistent novel adjacency predictions for 44 of the 511 deletions. The probabilistic method selects t-multiread alignments that are consistent with a single novel adjacency prediction, often removing incorrect alignments from the prediction. Examples are shown in Figures 4.5 and 4.6.

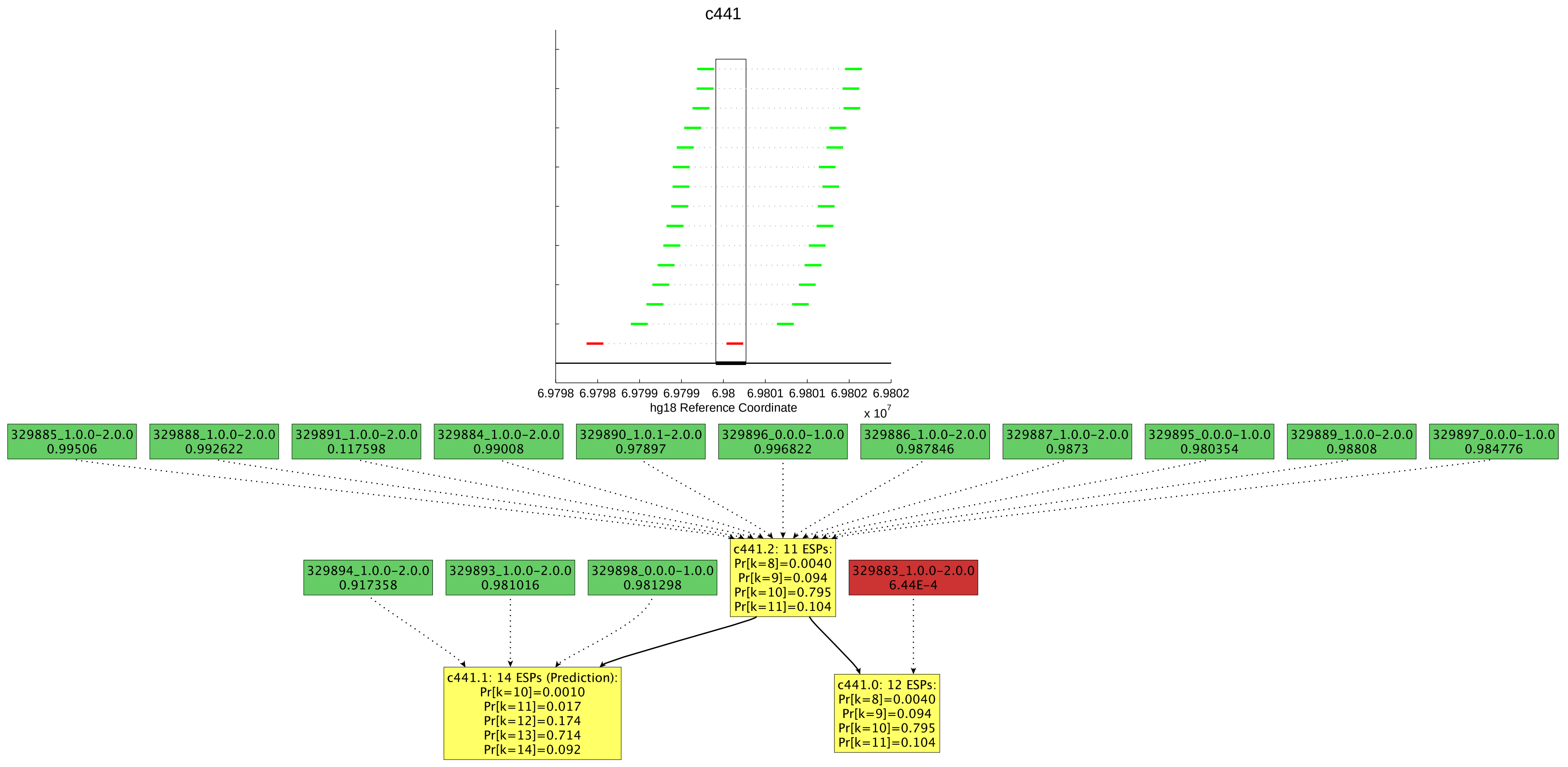

Figure 4.5: **Example c441 of a cluster diagram from 5X strobe dataset in the 1000G simulation.** (Top) 1-dimensional of the discordant pairs plotted along the hg18 reference. Green reads are correctly-aligned pairs, red reads are incorrectly-aligned pairs, and the black rectangle denotes the true deletion. (Bottom) Cluster diagram showing the correctly-aligned pairs (green), incorrectly-aligned pairs (red), and three candidate novel adjacencies (yellow). c441.14 is selected as the predicted novel adjacency by the probabilistic method, which includes all the correct alignments and excludes the single incorrect alignment.

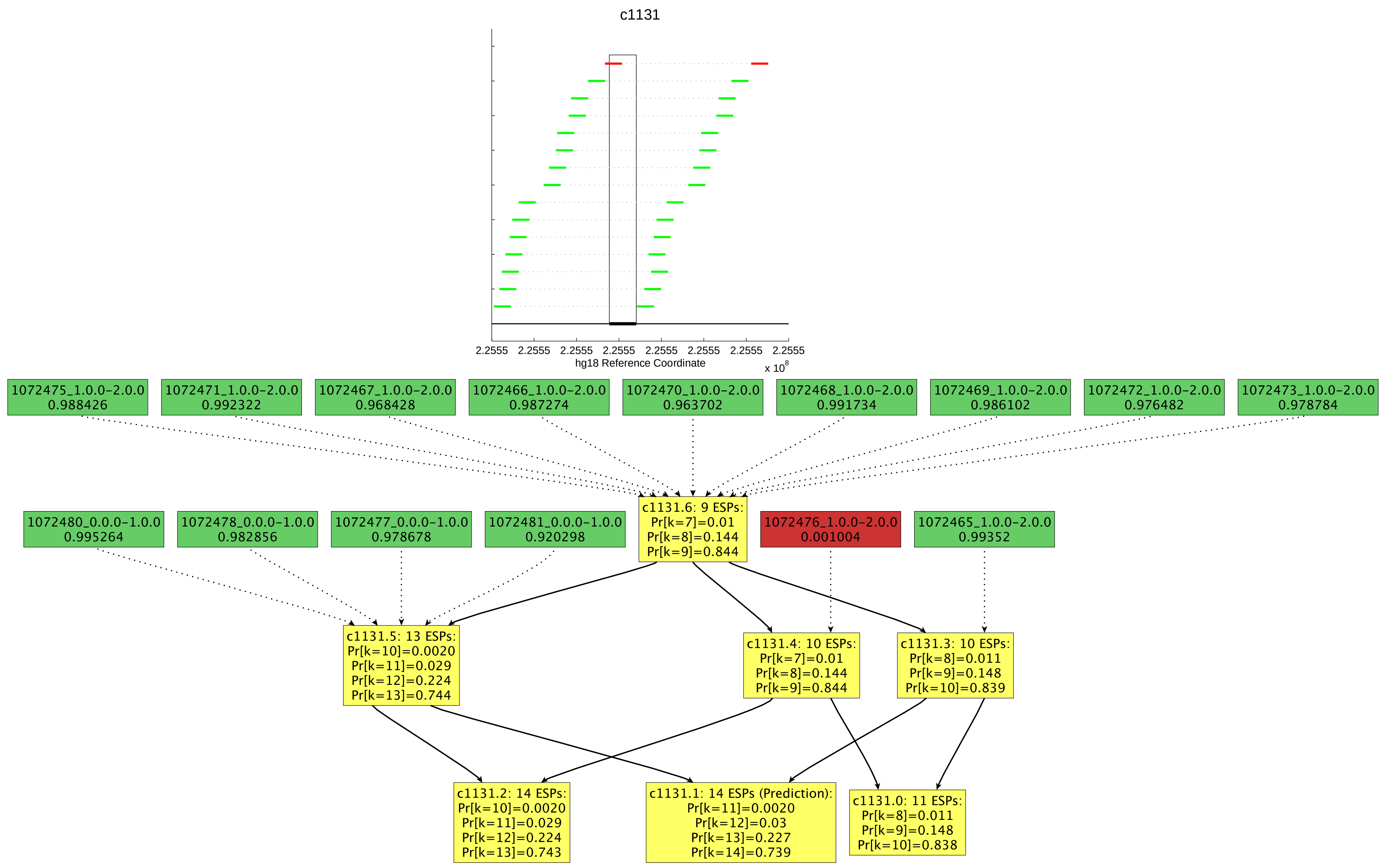
c1131
2.2555
hg18 Reference Coordinate
x 10^8
1072475_1.0.0-2.0.0 0.988426
1072471_1.0.0-2.0.0 0.992322
1072467_1.0.0-2.0.0 0.968428
1072466_1.0.0-2.0.0 0.987274
1072470_1.0.0-2.0.0 0.963702
1072468_1.0.0-2.0.0 0.991734
1072469_1.0.0-2.0.0 0.986102
1072472_1.0.0-2.0.0 0.976482
1072473_1.0.0-2.0.0 0.978784
1072480_0.0.0-1.0.0 0.995264
1072478_0.0.0-1.0.0 0.982856
1072477_0.0.0-1.0.0 0.978678
1072481_0.0.0-1.0.0 0.920298
c1131.6: 9 ESPs: Pr[k=7]=0.01 Pr[k=8]=0.144 Pr[k=9]=0.844
1072476_1.0.0-2.0.0 0.001004
1072465_1.0.0-2.0.0 0.99352
c1131.5: 13 ESPs: Pr[k=10]=0.0020 Pr[k=11]=0.029 Pr[k=12]=0.224 Pr[k=13]=0.744
c1131.4: 10 ESPs: Pr[k=7]=0.01 Pr[k=8]=0.144 Pr[k=9]=0.844
c1131.3: 10 ESPs: Pr[k=8]=0.011 Pr[k=9]=0.148 Pr[k=10]=0.839
c1131.2: 14 ESPs: Pr[k=10]=0.0020 Pr[k=11]=0.029 Pr[k=12]=0.224 Pr[k=13]=0.743
c1131.1: 14 ESPs (Prediction): Pr[k=11]=0.0020 Pr[k=12]=0.03 Pr[k=13]=0.227 Pr[k=14]=0.739
c1131.0: 11 ESPs: Pr[k=8]=0.011 Pr[k=9]=0.148 Pr[k=10]=0.838

Figure 4.6: **Example c1131 of a cluster diagram from 5X strobe dataset in the 1000G simulation.** (Top) 1-dimensional of the discordant pairs plotted along the hg18 reference. Green reads are correctly-aligned pairs, red reads are incorrectly-aligned pairs, and the black rectangle denotes the true deletion. (Bottom) Cluster diagram showing the correctly-aligned pairs (green), incorrectly-aligned pairs (red), and seven candidate novel adjacencies (yellow).

Venter Simulation

Following other methods [26, 124, 134], we derived a test genome by including approximately 17,000 adjacencies (including deletions, insertions, and inversions) from HuRef into hg18 chr17 [73]. From these rearrangements, 124 deletions greater than 120bp are used as our set of detectable deletions. Of the 124 deletions, 112 (90.32%) have repetitive sequence spanning both of the novel adjacency coordinates. Further, the inserted adjacencies consist mostly of small ($<$ 20bp) insertions and deletions, which tend to "shift" novel adjacency prediction coordinates.

As in the 1000 Genomes simulation, the probabilistic method improves sensitivity and specificity over the parsimony method (Figure 4.7). In variant calling, the 5X strobe dataset outperforms all other datasets when run with the probabilistic method, followed by the hybrid dataset (Figure 4.7). In fact, both 2X and 5X strobe datasets perform better at all sensitivities than the paired dataset regardless of the method. In alignment accuracy, the probabilistic method slightly enriches for correctly-assigned alignments in the pairs dataset compared to other methods (Figure 4.7). As before, the strobe datasets are consistently more enriched for correctly-assigned alignments than pairs.

The poor relative performance of the pairs dataset is caused by the increased difficulty of the Venter simulation as it relates to short read alignment. First, many true positive alignments in the Venter dataset are removed during the *initial* alignment stage. The BWA aligner, when run in the paired-mode, prefers to find t-multiread alignments that are concordant. Thus, due to the repetitive sequence at the novel adjacency coordinates, BWA mistakenly reports an *incorrect* concordant alignment for 47.9% of the t-multireads that support a deletion in the Venter test chromosome compared to 10.0% in the 1000 Genomes simulation. There are also more false positive alignments, 24.9% for Venter compared to 9.8% for 1000 Genomes, derived from more varied sources of error (Table 4.1).

Lastly, we assessed the ability to recover the four inversions in the Venter simulation (Table 4.2). While the 2X strobe dataset captures the four inversions, it also produces 50 false positives. The paired dataset, on the other hand, predicts two of the inversions with no false positives, but cannot detect the other two. When we combine the two datasets, the hybrid dataset detects all four inversions with six false positives. The 5X strobe dataset does an even better job at predicting inversions, requiring only three false positives to recover the four inversions.

MCMC Convergence

As we mention above, the solution space of all possible mappings is extremely large; however, we are able to divide the solution space in such a way that we can run the MCMC method on independent subproblems. Still, the question remains as to whether the method has converged on the subproblems. We performed two analyses to assess the convergence, which we will describe in terms of the 2X strobe dataset, which decomposes into 193 independent subproblems.

First, for the states that are sampled in the Markov chain, we have the frequency that the states were

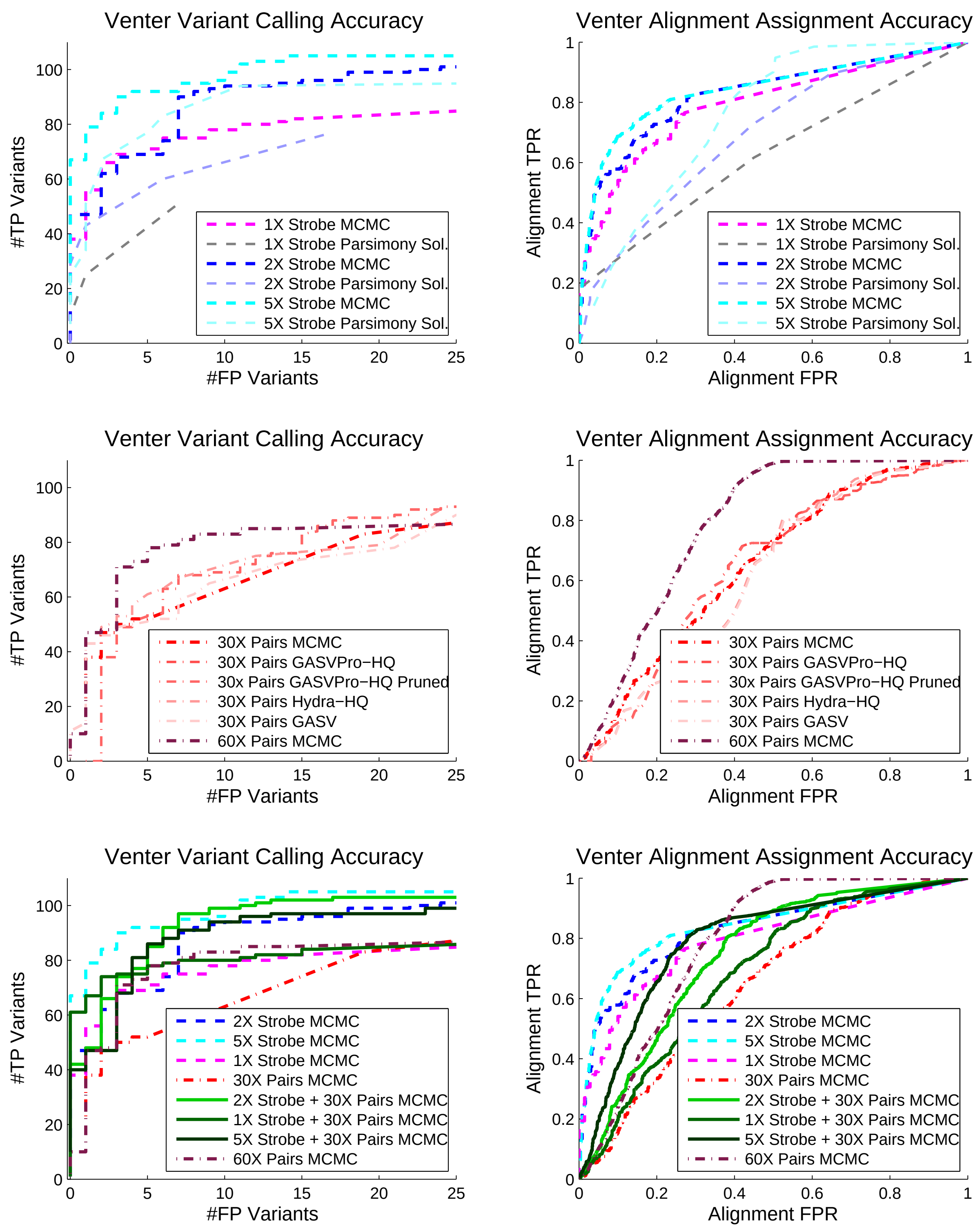

Figure 4.7: **(Left) Variant calling accuracy and (Right) alignment assignment accuracy for datasets from the Venter simulation.** Dashed lines are strobe datasets, dot-dash lines are paired datasets, and solid lines are hybrid datasets. The plots are plotted as (top) MCMC vs. Parsimony Solution for 1X,2X, and 5X strobes, (middle) the paired dataset run with a variety of methods, and (bottom) all the datasets run with the probabilistic method. Boldface lines are datasets run with our model: we compared the strobe datasets to a parsimony solution [124] and the paired datasets to Hydra [119], GASV [133], and GASVPro [134]. GASVPro-HQ is the original predictions output by GASVPro on unique alignments, and GASVPro-HQ Pruned removes overlapping variants from the predictions. Hydra-HQ is the Hydra method applied to unique alignments.

sampled and their (unnormalized) posterior probability. If the chain has converged, then the relative frequencies of these states should correspond to the relative posterior probabilities. Second, if we initialize the MCMC algorithm to start at the *correct* solution, then these relative frequencies should reflect the randomly-initializes relative frequencies. Thus, we have three sets of values that should be similar relative to each other. Figure 4.8 shows these values for three subproblems (out of 193 subproblems) we encounter for the 2X Strobe dataset. The small subproblem has four strobes with 16 possible solutions; the medium-sized subproblem has nine strobes with 512 possible solutions, and the largest subproblem, set 27, has 29 pairs from three different strobes with a total of $55,296$ possible solutions. We ran the largest subproblem for five million iterations, and we sample 72 of these solutions. The smallest probability (normalized across the 72 states) is 3.1779×10^{-4}, which is an upper bound on this states' true probability. The sampled states begin to deviate from the relative ordering of the probabilities when the probabilities are lower than 8.5×10^{-3}, and the first time the correctly-initialized states and the randomly-initialized states differ is when the state probability is less than 1.9×10^{-3}.

4.3.2 Sequenced Fosmids from Individual NA15510

In collaboration with Pacific Biosciences, we have sequenced four fosmids that contain reported SVs.

Fosmid Selection

To select fosmids for sequencing, we first evaluated the ability for next-generation paired read sequencing data to detect reported SVs in 63 fully-sequenced fosmids (44 deletions and 19 inversions) sequenced from individual NA15510 [62]. To assess the ability for paired-reads and strobes to detect the reported SVs, we simulated 30X coverage of paired-reads and strobes and counted the number of SVs that were detectable by the simulated data (Table 4.3). An SV is *detectable* if a cluster with at least five supporting discordant pairs lies within fosmid's coordinates when aligned to the reference.[2]

We selected two deletions detectable by both datasets as controls and two inversions that were detectable by strobes but not paired-end reads as "difficult" cases (Figure 4.9). The reported breakpoints of the inversions are flanked by segmental duplications; I1 contains segmental duplications with 95% sequence similarity, while I2 contains segmental duplications with 99% sequence similarity. Inversions often appear within such segmental duplications, making their detectability difficult (Figure 4.9 Bottom).

Fosmid Statistics

We obtained strobe sequencing data from Pacific Biosciences on the four fosmids harboring the selected SVs. The 3-strobes were sequenced to 14X-54X coverage (Figure 4.10). The inversion datasets are about

[2]Since the fosmids harbor SVs, a full alignment to the reference is not always obtainable. Thus, we find the best-scoring partial alignment (according to BLAST) and add a buffer of 100Kb. Since this region is much larger than the fosmid length, the detectability counts are conservative in the sense they may include false positive calls.

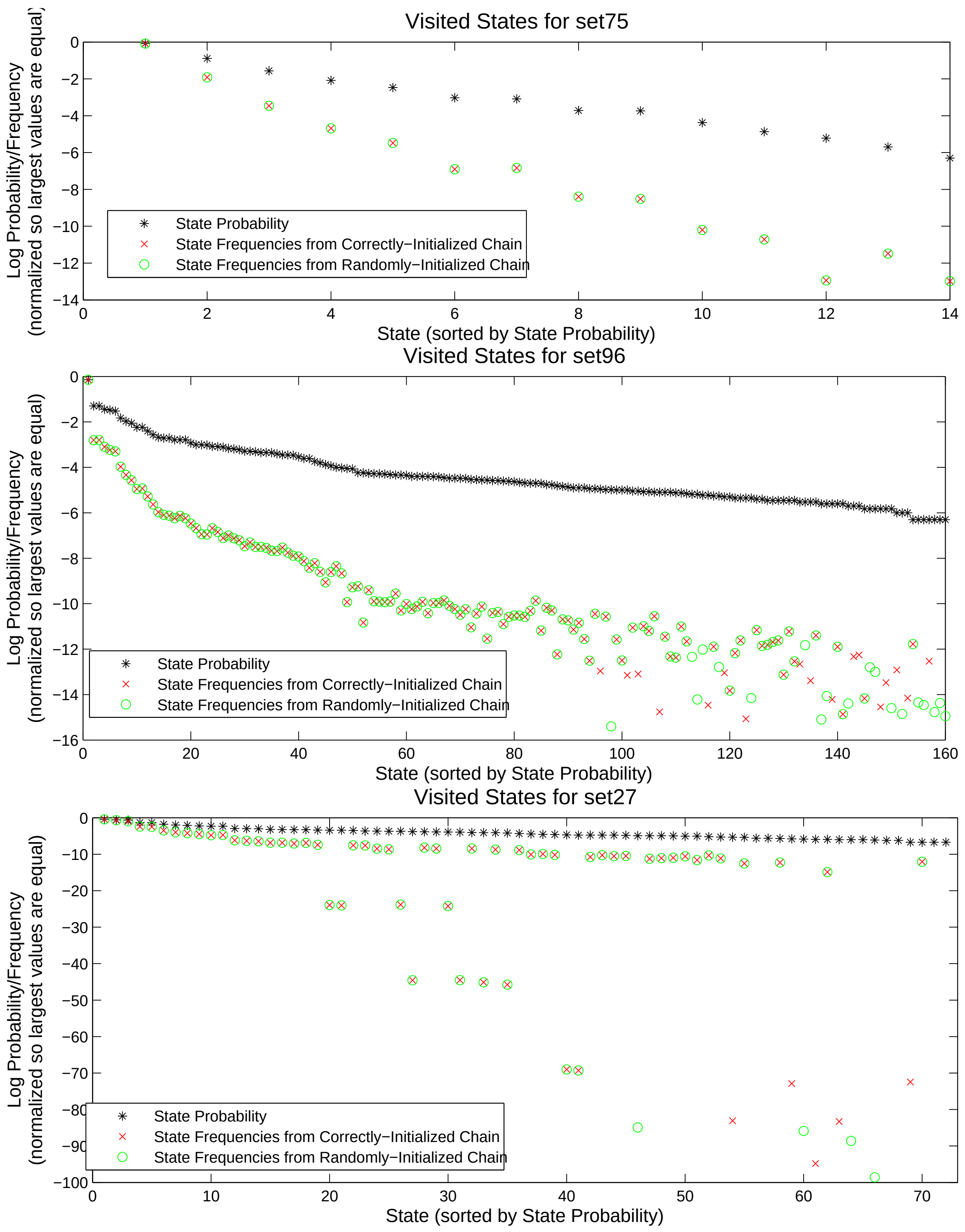

Figure 4.8: **MCMC Convergence on Subproblems for 2X Strobes.** MCMC convergence on a small subproblem (Top), an average-sized problem (Middle), and the largest subproblem (Bottom). The black asterisks denote the (unnormalized) probabilities of each state sampled in the 5 million iterations of the MCMC method. The red crosses and the green circles denote the relative sampling frequencies of each state for the correctly-initialized and randomly-initialized chains.

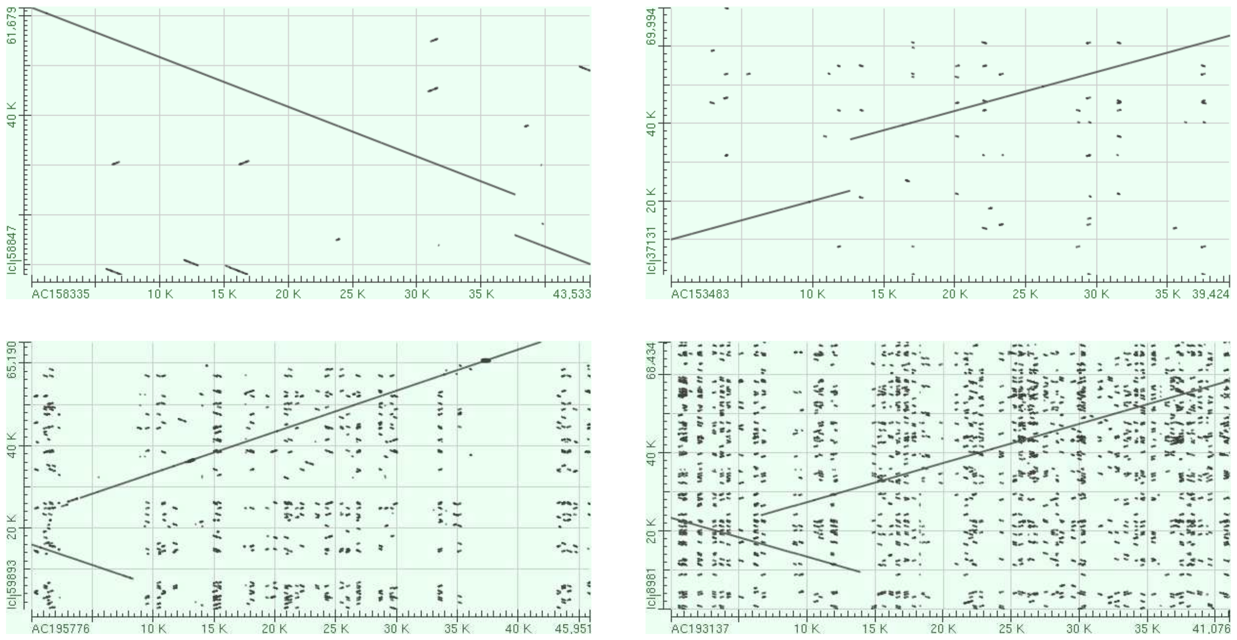

Figure 4.9: **Dot Plots for the Four Sequenced Fosmids.** Plots showing the sequence similarity for (Top Left) D1, (Top Right) D2, (Bottom Left) I1, and (Bottom Right) I2. Each plot shows the sequence similarity of the fosmid (x-axis) aligned to hg19 (y-axis). D1 is from the revers-complement strand with respect to the reference genome.

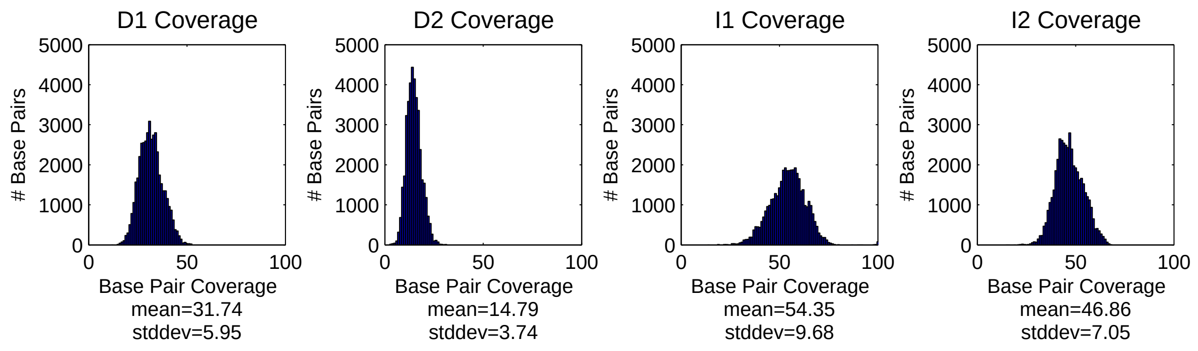

Figure 4.10: **Sequence coverage for the four fosmids.** Sequence coverage is determined by aligning strobes back to the fosmid sequence using BLASR and counting the number of times each base is covered by an alignment.

twice as large as the deletion datasets because two movies were taken for the inversion fosmids, compared to one movie for the deletion fosmids. We aligned the sequenced 3-strobes to hg19 using BLASR, Pacific Biosciences' in-house aligner, and ran the MCMC method on the resulting alignments. When the method has run to completion, we have a probability that each candidate SV is supported by k discordant pairs (see Methods). An inferred SV is considered a true positive if the discordant pairs imply an SV breakpoint within the junction range of the reported breakpoint in Table 4.4, otherwise it is a false positive.

Fosmid Results

To run the MCMC method on the fosmid datasets, we set the sequencing error rate $p_{seq} = 0.15$ and varied two user-defined input parameters: λ_d, the expected number of discordant pairs that support an SV, and p_{err}, the fixed probability that the correct strobe alignment is not present in the set of possible strobe alignments. Results are shown in Table 4.5 and Figure 4.11.

The MCMC method infers the correct SV for all parameter choices for both of the deletions (D1 and D2). Additionally, there are zero false positives for all parameter choices for D1 and when λ_d (the expected number of supporting discordant pairs) is large enough for D2. In the worst case, a single false positive is predicted for D2.

For inversion I1, the true positive prediction is found when the combination of parameters are small enough (in 9 out of 20 runs). Further, I1 infers a number of false positives, reaching 9 false positives at the smallest choices for λ_d and p_{err}. For parameter choices where the true positive is found, the average number of false positives for I1 is 5.89. The MCMC method infers the correct SV for all parameter choices for inversion I2, however, and returns an average of 0.65 false positives. It is surprising that the inversion in I2, with 99% sequence similarity, is easier to detect than the inversion in I1, with 95% sequence similarity. Upon closer inspection of this segmental duplication, we found that the lengths of the segmental duplications differed in I1, resulting in a lower sequence similarity. Partial alignments of the segmental duplications are 97% identical.

4.4 Discussion

In this chapter, we presented a probabilistic framework for SV detection for strobe sequencing data. The method works with previous, paired-read technologies, as well as seamlessly integrates multiple data types. It is robust to the sequencing error rate, and explicitly models the quality of the alignments in the probabilistic framework.

We show that, in simulation, on highly-repetitive datasets combining strobes with paired-end sequencing data improves performance. Further, on datasets where paired-end sequencing performs well, the MCMC method infers more alignments that are correct when there is enough signal from the alignment quality to differentiate the correct mappings from the incorrect mappings. The method outperforms previous methods

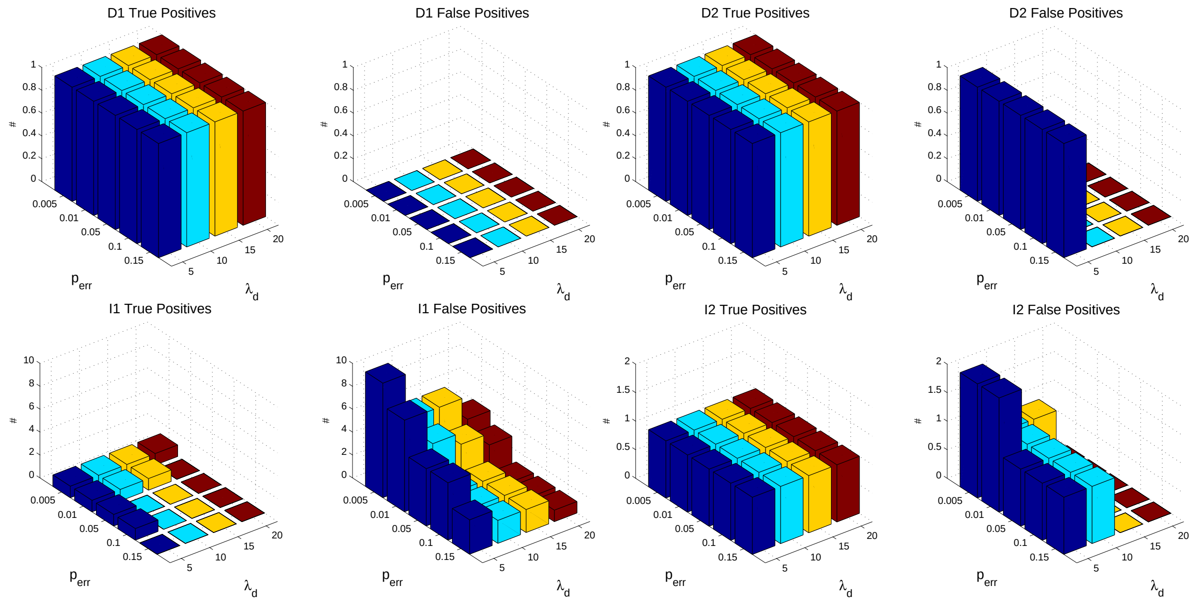

Figure 4.11: **Visual Representation of MCMC Results on Sequenced Fosmids.** For each fosmid, (Top Left)D1, (Top Right) D2, (Bottom Left) I1, (Bottom Left) and I2, the plot on the left indicates the number of true positives recorded and the plot on the right indicates the number of false positives for combinations of $\lambda_d = (5, 10, 15, 20)$ and $p_{err} = (0.005, 0.01, 0.05, 0.1, 0.15)$. The true positive and false positive plots are scaled by the same range on the Z-axis for each fosmid.

for strobe sequencing data by decreasing the number of inferred false positives, both in terms of the structural variants predicted and in terms of the alignments.

Finally, we applied our method to real strobe sequencing data from four sequenced fosmids: two fosmids contained deletions and were considered controls and two fosmids contained inversions that were impossible to detect from simulated paired-read datasets due to flanking segmental duplications. We correctly predict the true positive SV in all cases, but we also predict a number of false positives in the inversion datasets.

We note that our method, while designed for strobe sequencing, has a much broader range of impact for SV prediction. One example is the application of our method for split reads - as sequenced reads become longer, these reads may harbor multiple structural variants. Current split read mappers such as Pindel [155] could produce candidate alignments of long reads represented as high-quality alignments with gaps; our framework would allow candidate alignments that contained multiple gaps, producing a “virtual strobe” with linked substrings. Further, our method is general enough to handle paired-end sequencing data as well, allowing a method for cross-platform analysis.

There are some limitations to our method; in particular, it has been shown in [134] that there is significant signal in the read depth from concordant pairs, which we completely disregard. This read depth signal further reduces the number of false positives, particularly for copy number variants. An additional improvement to this model would be to incorporate this type of information, in addition to the alignment quality and expected support.

Finally, we note that in highly-repetitive regions, sometimes there is not enough signal in the alignment quality to be of much use; in these cases, our method performs as well on paired-end data as other methods. In the case where there is additional signal in the alignment quality (i.e. the correct alignments align with the sequencing error we expect, and incorrect alignments align with worse sequencing error) or method incorporates this information in a useful manner.

	1000GChr1	1000GChr17	VenterChr17	VenterSimple
Test Chromosome Construction				
Chr	1	17	17	17
# insertions	557	57	8752	389
# deletions	237	218	8801	425
# inversions	0	0	4	4
Total # of SVs	794	275	17377	822
# deletions $\geq$ 120bp (**D**)	511	197	124	124
Simulated Paired Data				
# of fragments that span a del. in **D** (**S**)	16679	6020	3466	3585
Avg. # of fragments that span a deletion	32.6	30.5	29.5	28.9
BWA Alignments				
# of deletion ESPs $>$ 600bp and $<$ 1Mb	15300	5371	1791	1845
– # with correct coordinates (**TP**)	13802 (90.2%)	4695 (87.4%)	1345 (75.1%)	1446 (78.3%)
– # with incorrect coordinates (**FP**)	1498 (9.8%)	676 (12.6%)	446 (24.9%)	399 (21.6%)
# in **D** supported by at least one **TP**	499	191	93	88
Where do FPs Come From?				
–# from split read	1440 (96.1%)	674 (99.7%)	284 (63.7%)	261 (65.4%)
–# from read in novel sequence	55 (3.7%)	0	123 (27.6%)	107 (26.7%)
–# that span another SV	1	1	39 (8.7%)	31 (7.8%)
–# other	1	1	0	0
BWA Alignments for Fragments in S				
# with a deletion ESP alignment	13606 (81.6%)	4631 (76.9%)	1351 (39.0%)	1429 (39.9%)
– # with correct coordinates	13570 (99.7%)	4630 (99.98%)	1304 (96.5%)	1395 (97.6%)
# with a non-del. ESP alignment	22	3	16	19
# that are unmapped/qual $<$ 10	1392 (8.4%)	390 (6.4%)	639 (18.4%)	684 (19.1%)
# with a concordant alignment	1659 (10.0%)	996 (16.5%)	1659 (47.9%)	1452 (40.5%)
–# that are $\leq$ 600bp (truly concordant)	1120 (67.5%)	730 (73.2%)	663 (40.0%)	673 (46.4%)
–# that are $>$ 600bp	537 (32.4%)	195 (19.6%)	716 (43.2%)	707 (48.7%)
–# with read in novel sequence	2	71 (7.1%)	81 (4.9%)	72 (5.0%)

Table 4.1: **Simulation Statistics for 1000 Genomes and Venter simulations.** (1000GChr1) Original 1000 Genomes simulation as described in the main document. (1000GChr17) 1000 Genomes simulation applied to Chr 17, inserting 57 insertions and 218 deletions for 197 detectable deletions $\geq$120bp. (VenterChr17) Original Venter simulation as described in the main document. (VenterSimple) Venter simulation with variants >20bp applied to the test chromosome. Percentages are shown in parentheses when the value is $> 1\%$. The **TP** and **FP** alignments are determined by the alignment assignment accuracy.

	Inversion 1		Inversion 2		Inversion 3		Inversion 4	
	5826739-5827291		40566233-40567384		55552838-55556395		57999778-58000250	
	Prob	# FPs	Prob	# FPs	Prob	# FPs	Prob	# FPs
2X Strobes	0.002	6	1.000	0	0.921	1	0.00	50
5X Strobes	1.000	0	1.000	0	1.000	0	0.365	3
30X Pairs	NaN	NaN	1.000	0	NaN	NaN	1.000	0
Hybrid	0.001	6	1.000	0	0.479	1	0.500	0

Table 4.2: **Inversion Results for the Venter Simulation.** Four inversions predicted by the probabilistic method in the Venter simulation on 2X strobes, 5X strobes, 30X pairs, and a 2X strobes + 30X pairs hybrid dataset. Inversion coordinates refer to hg18 Chr17. The inversion probability and number of false positives incurred to detect the inversion are reported. NaN indicates that the inversion was not in the candidate input set; thus it could never be predicted.

	Simulations for 44 Deletions			**Simulations for 19 Inversions**		
	Paired-End Detected	Paired-End Undetected	Total	Paired-End Detected	Paired-End Undetected	Total
Strobe Detected	25	10	35	4	9	13
Strobe Undetected	2	7	9	0	6	6
Total	27	17	44	4	15	19

Table 4.3: **Detectability of Reported Adjacencies on Simulated Strobe and Paired-Read Data.** (Left) Over half of the reported deletions are detectable by both paired-end and strobe datasets; we selected two of the fosmids harboring these deletions as controls. (Right) Strobes are able to detect nine more inversions than the paired-end dataset; we selected two of the fosmids harboring these deletions as "difficult" cases.

Name	Accession	Chr	Reported Start [63]*	Reported End [63]*	Reported Junction [63]	Paired-Read Support	Strobe Support
D1	AC158335	Chr3	68739688	68747866	2	74	19
D2	AC153483	Chr16	78371638	78384899	0	84	15
I1	AC195776	Chr19	39264278	39280958	1236	0	21
I2	AC193137	Chr14	35017063	35031477	7063	0	13

*Coordinates lifted over from hg18 to hg19.

Table 4.4: **Fosmids Selected for Pacific Biosciences Sequencing.** The reported junction is the length of unmatched sequence found across the SV [63]. The last two columns report the number of supporting discordant pairs (or consecutive subread pairs) of the detectable SV in simulations.

Name	Sequence Coverage	$\lambda_d = 10, p_{err} = 0.01$ TP	FP	Range of λ_d and p_{err} * TP	FP
D1	31.74 ± 5.95	1	0	1 ± 0.00	0 ± 0.00
D2	14.79 ± 3.74	1	0	1 ± 0.00	0.25 ± 0.44
I1	54.35 ± 9.68	1	4	0.45 ± 0.51	3.7 ± 2.45
I2	46.86 ± 7.05	1	1	1 ± 0.00	0.65 ± 0.67

* $\lambda_d = \{5, 10, 15, 20\}$ and $p_{err} = \{0.005, 0.01, 0.05, 0.1, 0.15\}$

Table 4.5: **MCMC Results on Sequenced Fosmids.** For each fosmid (D1, D2, I1, and I2) multiple values of λ and p_{err} were simulated. 'TP' and 'FP' correspond to the number of 'True Positives' and 'False Positives', respectively.

Chapter 5

De novo Assembly using Strobe Sequencing

In the previous chapters, we have described methods for SV detection from strobe sequencing data using a resequencing approach, where we align the subreads to a known reference. However, there are many scenarios where the reference may be unavailable; for example, if we sequence the genome of an organism for which the reference may not have been determined yet. In the framework of sequencing human genomes, some genomes have undergone significant rearrangement (as in the case of cancer), and some regions of the reference may not be useful for determining structural variation. Further, there is a certain amount of bias in the reference genome, which might affect the SV prediction calls. Thus, *de novo* assembly of reads, which does not require a reference genome, becomes an attractive option.

In this chapter, we introduce a method for small-scale genome assembly using strobe sequencing data. Many of the existing assembly methods (which are summarized below) under-utilize a key piece of information when constructing the graph - the pairing information. Since strobes have more subreads than mate pairs, this information is more powerful for strobes. All the graph-based methods incorporated pairing information after the graphs are constructed; instead, we wish to simultaneously use the overlap information and the pairing information through the assembly method.

We propose a novel algorithm for small-scale *de novo* assembly from strobes that use both the sequences of the subreads *and* information about the length of advances between subreads at the same time when building the assembly. In brief, we determine a set of linear constraints that represent the relationships between subreads from the same strobe, and we formulate the assembly problem as an optimization problem that aims to maximize the expected pairwise overlaps while respecting these constraints.

5.1 Related Work

In general, assemblers construct one of two types of graphs. Overlap graphs [50, 96, 135] are graphs where the vertices are reads and an edge exists between two vertices if the corresponding reads align to each other. Overlap graphs are the first component of "overlap-layout-consensus" algorithms, where the overlap graph is constructed, graph simplifications are performed, a path through this graph determines a layout of the reads, and the sequence is determined from the resulting layout. The size of overlap graphs scales linearly with the number of reads, prohibiting whole-genome assemblies.

An alternative graph construction that has been considered is the de Bruijn graph [110, 111]. All strings of length k (k-mers) are noted in the set of reads, and the vertices of a de Bruijn graph are the $(k-1)$-mers. An edge connects two vertices if there exists a k-mer that contains the two $(k-1)$-mers. Currently, de Bruijn graph assemblers [23, 24, 76, 86, 132] hold the most promise for handling large assemblies, and two reported whole-genome assemblies used de Bruijn assemblers [76, 132].

The Euler algorithms (EULER [110, 111], EULER-SR [24] and EULER-USR [23]) first correct sequencing errors in reads by applying a filter to the reads before building the de Bruijn graph. They then layer pairing information after the de Bruijn graph is constructed and choose paths in the graph according to the pairing information in order to resolve repeats. These algorithms include the construction of the A Bruijn graph [109], where nodes refer to consensus sequences from multiple sequence alignments rather than k-mers from individual reads. Velvet [159, 160] performs many types of graph simplifications on the de Bruijn graph in an iterative fashion. In addition to tip removal (similar to a removal algorithm performed by EULER) and bubble removal, it also removes paths that have low coverage in terms of the number reads each path indicates. Velvet also incorporates mate pair information, first in an algorithm called breadcrumb [159] and later in using an algorithm called pebble [160]. ALLPATHS [17, 45] uses a de Bruijn graph framework, but instead of building a graph representing a global assembly of the reads it instead builds local assemblies from a set of seeds. The local assemblies are then glued together using information from all paths between each pair of reads. In a similar vein, SHRAP [138] uses a hierarchical sequencing protocol that identifies sets of reads to assemble from the k-mer content of each read and then uses EULER for the assembly. [86] presents the first exact polynomial-time algorithm for a double-stranded genome by first estimates copy counts with mate pair data, then uses a maximum likelihood approach to assemble the reads into contigs and finally extends the contigs using pairing information.

Few of the de Bruijn assemblers are capable of assembling large genomes such as human genomes due to large space and memory requirements. Only two de Bruijn based methods, SOAPdenovo [76] and ABySS [132], have reported whole human genome assemblies. SOAPdenovo [76] uses a similar approach as Velvet, but does not record all the information about the read locations and paired end locations in the graph. This makes SOAPdenovo quite space-efficient, and able to handle larger amounts of data. ABySS [132] implements a distributed de Bruijn graph, making the assembler memory-efficient. However, both methods report highly fragmented assemblies: SOAPdenovo assembles only 85% of an African genome and 87% of an Asian genome, and ABySS reports over 680,000 contigs $\geq$ 1Kb for an African genome.

Higher error rates, such as those found in Pacific Biosciences technologies, introduce further complications. Errors in the reads introduce erroneous edges in overlap graphs and de Bruijn graphs, and in the de Bruijn graphs the seed size k is dependent on the sequencing error rate. In most assemblers, extra error-correction steps are taken before, during, and after the graph construction phases. Repeats in the target genome are collapsed in the graph constructions, and result in ambiguous path choices when determining the read layout. Additional use of paired-end sequencing helps identify paths in the graph by linking nodes using the pairing information.

Finally, we note that there are a number of assemblers that differ from the overlap graph or de Bruijn graph approaches. PE-Assembler [4], SSAKE [151], QSRA [16], and VCAKE [59] use greedy methods to extend the 3' end of the high-quality read sequences. [121] presents a different graph, the alignment graph, that represents a multi-read alignment, which is used in a consensus algorithm for assembly. Genovo [71] presents a probabilistic model for the generation and sequencing of fragments and applies the model to metagenomic data. While this is a different application (the goal with metagenomic data is to construct assemblies of *multiple* organisms simultaneously), Genovo has a number of attractive qualities. First, it defines a generative probabilistic model of read generation from multiple organisms and second, it discovers the most likely sequence of reconstructions under this model. However, it is designed for paired-end sequencing data (as are all methods described above).

Current assembly algorithms do not use information from paired reads until after building an overlap graph from individual reads. A similar approach could be used for t-strobes by constructing an overlap graph from individual subreads and then finding paths in the graph that simultaneously satisfy the length constraints between adjacent subreads. However, there are two downsides to this approach. First, because of high per-nucleotide error rates (5-15%), an overlap/deBruijn graph constructed from individual subreads of a strobe would have a large number of erroneous edges from spurious alignments. Second, the number of possible paths to explore would grow enormously in repetitive regions, and even more so because of erroneous edges. Aggressive heuristics designed to reduce the size of the search space would likely reduce any advantage for strobes in resolving repetitive regions. Our approach aims to avoid these problems.

5.2 A Combinatorial Optimization Method for Small-Scale Assembly with Strobes

Our method incorporates pairwise subread overlap information and strobe constraint information simultaneously to assemble a set of strobes. To do so, we design an Integer Linear Program (ILP) that properly constrains the subreads in an assembly while including as many high-scoring pairwise overlaps as possible. While the number of integer variables scales linearly with the number of subreads, the number of binary variables scales quadratically with the number of subreads. Thus, we iteratively build an assembly by assembling subsets of strobes.

We first present the general assembly problem for a set of reads. The reads are modeled as substrings of an original, unknown string G (which we will also call a *genome* G) with insertions, deletions and substitutions introduced according to some error model. The number of differences between the read and the original strings is bounded by the error rate. The task of genome assembly is to recover the substring coordinates from the original string using the reads.

Consider a set $\mathbf{R}$ of reads from a genome G, where each read $R \in \mathbf{R}$ is a sequence of bases from an alphabet Σ (typically $\Sigma = \{A, C, G, T\}$) and the number of measured bases for read R is denoted by $|R|$. We define an error function $\varepsilon(r)$ that determines the maximal number of base pairs that may be inserted or deleted in a read of length r. A *layout* $\mathcal{L}_{\mathbf{R}}$ of the reads $\mathbf{R}$ is a set of $|\mathbf{R}|$ pairs of integers

$$\mathcal{L}_{\mathbf{R}} = \{(x_R, y_R) : |R| - \varepsilon(|R|) \le |y_R - x_R| \le |R| + \varepsilon(|R|)\}_{\mathbf{R}} \tag{5.1}$$

that give the coordinates of each read $R \in \mathbf{R}$. Note that the coordinates must be within $\varepsilon(|R|)$ of the number of measured bases in R, denoting at most $\varepsilon(|R|)$ insertions or deletions. Additionally, the (x, y) pair for each subread R provides the subread orientation: if $x < y$, then R was generated from the target string. Otherwise the reverse complement of R was generated from the target string (denoted by R^c). Thus, a layout is a set of signed intervals on the number line. These intervals need not cover the entire line; rather, there may be sets of intervals that collectively cover line segments. We call these sets of intervals *contigs*.

Two subreads R and R' *overlap* in the layout $\mathcal{L}_{\mathbf{R}}$ if the intervals defined by the coordinates (x_R, y_R) and $(x_{R'}, y_{R'})$ have a non-empty intersection. Given an overlap amount δ, we say that R and R' $delta$-overlap if the magnitude of their intersection is at least δ.

The Sequence Assembly Problem. Given reads $\mathbf{R}$ from a genome G, find a layout $\mathcal{L}^*_{\mathbf{R}}$ that maximizes some layout score $Q(\mathcal{L}^*_{\mathbf{R}})$.

A layout $\mathcal{L}_{\mathbf{R}}$ may be scored in any number of ways [48]. For instance, one may derive the target string from the layout (a step known as *consensus building*) and assess the similarity of each read R and the substring from the original string determined by the coordinates (x_R, y_R). Alternatively, one may assess the similarities of all pairs of reads (R, R') whose intervals (x_R, y_R) and $(x_{R'}, y_{R'})$ have a nonempty intersection on the number line. We choose to score a layout using a pairwise similarity metric based on the alignments of subreads.

Computing Pairwise Alignments. A *global alignment* of two subreads $R, R' \in \mathbf{R}$ is a two-row matrix where the top row contains characters of R in sequential order and the bottom row contains characters of R' in sequential order, and spaces (represented as dashes) may be interspersed in the characters of R and R' such that no column contains two spaces [48, 61]. Given a score for each pair of characters in the string alphabet (including the space character), the scores for each column of an alignment are summed to produce an alignment score $\Phi(R, R')$. The Smith-Waterman algorithm is a dynamic program to compute the optimal alignment given a scoring matrix for all pairs of characters in the alphabet (including the space character).

An *end-space free alignment* of R and R' is a global alignment that does not score columns with spaces at the beginning or the end of the alignment [48]. End-space free alignments are utilized in assembly because the reads are expected to partially overlap each other (the prefix of one subread may overlap the suffix of another, for example). We use a trivial modification of the Smith-Waterman alignment to compute the optimal end-space free alignment for all pairs of subreads.

Since DNA is double stranded, the optimal end-space free alignment of R and R' might include their reverse complements, denoted by R^c and R'^c. Without loss of generality we assume that the first subread R is in the forward orientation. Thus, in addition to computing the end-space free alignment of (R, R') we also compute the optimal end-space free alignment of (R, R'^c) and choose the one with the best alignment score. Note that if R is aligned to R'^c, the second row of the alignment matrix contains the reverse complement of R' and the first character of R' appears as the last character in the second row. Regardless of the orientation of R', we call the score of the resulting alignment $\Phi(R, R')$.

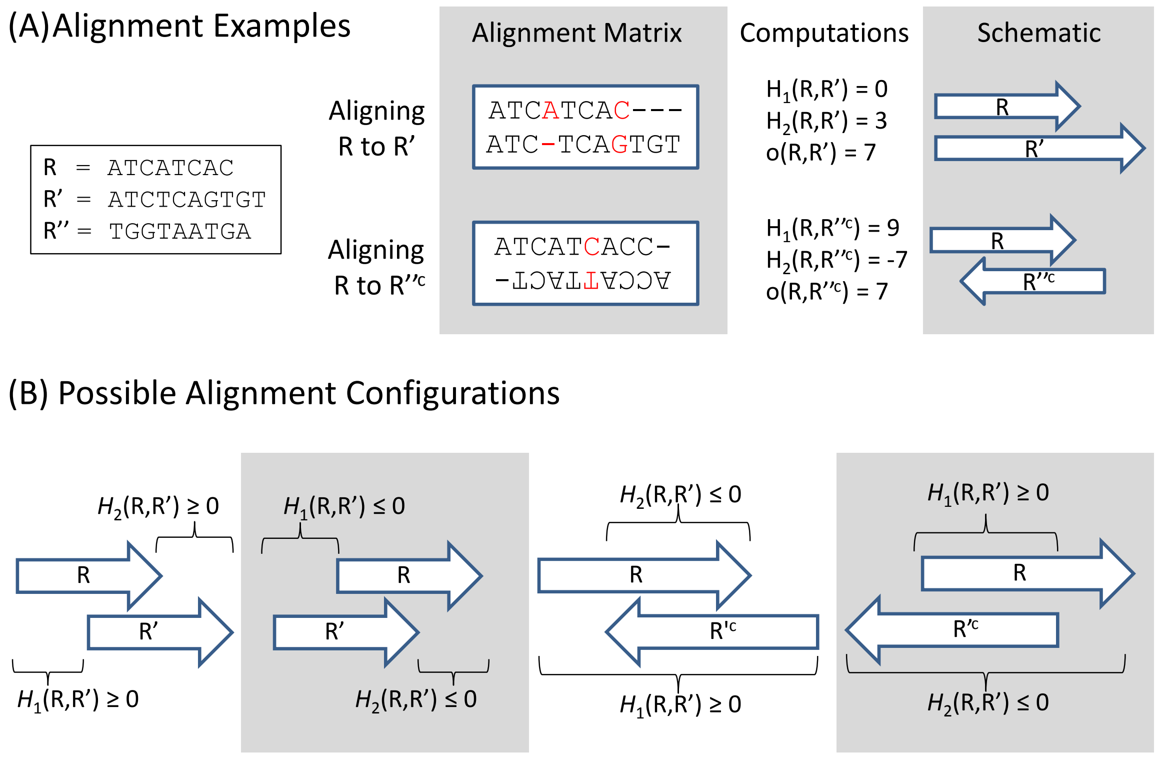

Figure 5.1: **Alignment Examples and possible alignment configurations.** (A) Two examples of end-space free alignments. The alignment of R and R' contains an indel and a substitution (in red in the alignment matrix). The alignment of R and R''^c contains a substitution. (B) $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ for possible alignment configurations of R and R'. Note that, if R and R' are in the same orientation, then the sign of $\mathcal{H}_1(R, R')$ is the same as the sign of $\mathcal{H}_2(R, R')$ (or they are both 0). If R' is in the reverse orientation, then one of $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ is non-negative and the other is negative.

From an optimal alignment of R and R', we define two alignment terms that describe their alignment configuration (Figure 5.1). In terms of the alignment matrix, $\mathcal{H}_1(R, R')$ is the difference between the column of the first character in R' and the column of the first character in R. $\mathcal{H}_2(R, R')$ is the difference between the column of the last character in R' and the column of the last character in R. Note that, while

$\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ are related, they are not symmetric. From $\mathcal{H}_1(R, R')$, $\mathcal{H}_2(R, R')$, and the measured number of bases $|R|$ and $|R'|$, the orientation of subread R' is determined as well as the substrings of R and R' that are aligned (with matches, mismatches, or indels). However, $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ do not uniquely determine the alignment matrix (i.e. there may be any number of gaps in the aligned portions of the substrings).

We also define $o(R, R')$ as the *aligned* portion of the alignment (Figure 5.1). That is, $o(R, R')$ is the difference of the largest column containing a starting or ending character in R or R' and the smallest column containing a starting or ending character in R or R'. We say that two subreads *δ-align* if $o(R, R') + 1 \geq \delta$. Note that while $\mathcal{H}_1(R, R')$, $\mathcal{H}_2(R, R')$, and $o(R, R')$ collectively indicate the number of spaces interspersed in the aligned portion, again they do not uniquely determine the alignment matrix.

Finally, we say that R *contains* R' if the first and last columns of the alignment matrix contain the first and last characters of R (in either order). Similarly, R' contains R if the first and last columns of the alignment matrix contain the first and last characters of R_j (in either order). If R and R' contain each other, then the end-space free alignment is identical to their optimal global alignment.

5.2.1 The Strobe Assembly Problem

Now, suppose *strobes* are generated from the genome G with sequencing error. Strobes are modeled as longer substrings of the target string with introduced sequencing error, and subreads are modeled as substrings of the strobe. Strobe notation is the same as §3.2; however, we removing the strobe indexing for clarity and work with the set of strobes.

Given a set $\mathbf{S}$ of t-strobes, we now define $\mathbf{R}$ to be the set of all subreads from strobes in $\mathbf{S}$. A *strobe layout* $\mathcal{L}_\mathbf{S}$ is a layout that respects the ordering, orientation, and pairing information between subreads on the same strobe.

$$\mathcal{L}_\mathbf{S} = \{(x_R, y_R)\}_\mathbf{R}$$

such that

$$\forall R \in \mathbf{R} : \quad |R| - \varepsilon(|R|) \leq |y_R - x_R| \leq |R| + \varepsilon(|R|) \tag{5.2}$$

$$\forall R_i, R_j \in S, S \in \mathbf{S} : \quad 0 < (y_{R_i} - x_{R_i})(y_{R_j} - x_{R_j}) \tag{5.3}$$

$$\forall R_i, R_{i+1} \in S, S \in \mathbf{S} : \quad \begin{cases} l \leq x_{R_{i+1}} - y_{R_i} \leq u & \text{if } x_{R_i} < y_{R_i} \\ l \leq -\left(x_{R_{i+1}} - y_{R_i}\right) \leq u & \text{otherwise} \end{cases} \tag{5.4}$$

These constraints imply that all subreads from the same strobe are in the same orientation, and the advance lengths between the subreads are within the proper bounds depending on the strobe orientation. The pairing information may indicate a relative ordering of contigs, called *scaffolds*, where a contig with no relative ordering to other contigs is itself a scaffold. Each scaffold in a layout thus consists of one or more contigs separated by strings of unknown bases with some lower and upper bound determined by pairing information of the strobes.

Note that the strobe layout prohibits subreads from the same strobe to overlap; thus, when computing the pairwise alignment scores for subreads $R \in S$ and $R' \in S'$, we only do so for subreads from different strobes ($S \neq S'$). We assign subreads from the same strobe an aligned portion of 0 (i.e. $o(R, R') = 0$ if $R, R' \in S$).

The Strobe Assembly Problem. Given a set $\mathbf{S}$ of t-strobes from a genome G, find a strobe layout $\mathcal{L}^*_{\mathbf{S}}$ that maximizes a layout score $Q(\mathcal{L}^*_{\mathbf{S}})$.

Scoring a strobe layout. We will define the strobe layout score $Q(\mathcal{L}_{\mathbf{S}})$ by using the end-space free alignments of the subread pairs. Since strobes are generated from a single genome string G, subreads that overlap in the strobe layout $\mathcal{L}_{\mathbf{S}}$ should align with high similarity for the aligned portion. Thus, if R and R' δ-align with a large alignment score $\Phi(R, R')$, we expect R and R' to δ-overlap in the layout $\mathcal{L}_{\mathbf{S}}$. We establish a set of *expected* δ-overlaps that have high alignment scores and a set of *unexpected* δ-overlaps that have low alignment scores. These are described in two binary matrices: an expected overlap matrix $\mathbf{O}$ and an unexpected overlap matrix $\mathbf{N}$. For $R, R' \in \mathbf{S}$, we define

$$O_{R,R'} = \begin{cases} 1 & \text{if } \Phi(R, R') \geq t_O \\ 0 & \text{otherwise,} \end{cases} \qquad N_{R,R'} = \begin{cases} 1 & \text{if } \Phi(R, R') < t_N \\ 0 & \text{otherwise} \end{cases}, \tag{5.5}$$

where t_O and t_N are score thresholds for an expected overlap and an unexpected overlap respectively ($t_N \leq t_O$). Note that there may be overlaps that have alignment scores between t_N and t_0; we call these *unspecified* overlaps. Since we do not compute overlaps between pairs of subreads R and R' from the same strobe, i.e. $R, R' \in S$, we trivially set $O_{R,R'} = 0$ and $N_{R,R'} = 1$.

Additionally, we need some terms to describe the strobe layout $\mathcal{L}_{\mathbf{S}}$ that we wish to score. We describe the subread relationships in $\mathcal{L}_{\mathbf{S}}$ using a binary overlap matrix β and an orientation vector η. For $S, S \in \mathbf{S}$ and $R \in S, R' \in S'$, we define

$$\beta_{R,R'} = \begin{cases} 1 & \text{if } R \text{ and } R' \delta\text{-overlap in } \mathcal{L}_{\mathbf{S}} \\ 0 & \text{otherwise.} \end{cases} \qquad \eta_S = \begin{cases} 1 & \text{if } x_R < y_R \,\forall\, R \in S \\ 0 & \text{otherwise.} \end{cases} \tag{5.6}$$

Note that, since a strobe layout requires the same orientation for all subreads in a strobe, η_i describes the orientation for the entire strobe. The layout score we define rewards expected overlaps, penalizes unexpected overlaps, and is indifferent to unspecified overlaps.

$$Q(\mathcal{L}_{\mathbf{S}}) = \sum_{R,R' \in \mathbf{S}} \underbrace{O_{R,R'}\beta_{R,R'}\Phi(R,R')}_{\text{reward expected overlaps}} - \underbrace{O_{R,R'}(1-\beta_{R,R'})\Phi(R,R')}_{\text{penalize missing overlaps}} - \underbrace{N_{R,R'}\beta_{R,R'}E[\Phi(R,R')]}_{\text{penalize unexpected overlaps}} \quad (5.7)$$

where $E[\Phi(R, R')]$ is the expected score of an alignment with an aligned portion of exactly $o(R, R')$. This can be computed analytically with the alignment scoring functions and the sequencing error rate ε or estimated empirically from the pairs of subreads of similar lengths whose aligned portions are $o(R, R')$.

5.2.2 Integer Linear Program (ILP) Formulation

We solve the strobe assembly problem by defining a set of linear constraints that determine a strobe layout and finding a set of position variables (x_R, y_R) for each subread R that maximize $Q(\mathcal{L}_{\mathbf{S}})$. We first describe each constraint, and then we show how to convert these constraints into linear constraints.

The constraints fall into roughly four groups, position constraints, advance constraints, overlapping constraints, and configuration constraints.

Position Constraints This group of constraints ensures that, for each subread R from strobe S, the position variables x_R and y_R are set such that the resulting layout is a strobe layout. First, all subreads from S must be in the same orientation, or the sign of $y_R - x_R$ must be the same for all $R \in S$. The orientation variable η_S determines the orientation for each subread in S, so we use η_S to ensure that the sign remains the same for all subreads $R \in S$:

$$\text{if } \eta_S = 1: \quad 0 < y_R - x_R \quad (5.8)$$

$$\text{if } \eta_S = 0: \quad 0 < -(y_R - x_R) \quad (5.9)$$

Second, each subread R must have at most $\varepsilon(|R|)$ indels. The distance between x_R and y_R must reflect this, or

$$\text{if } \eta_S = 1: \quad |R| - \varepsilon(|R|) \leq (y_R - x_R + 1) \leq |R| + \varepsilon(|R|) \quad (5.10)$$

$$\text{if } \eta_S = 0: \quad |R| - \varepsilon(|R|) \leq (-(y_R - x_R) + 1) \leq |R| + \varepsilon(|R|) \quad (5.11)$$

Advance Constraints These constraints ensure that the distance between adjacent subreads $R_i \in S$ and $R_{i+1} \in S$ are within the specified lower and upper bounds. Again, we use the η_S variable to determine the proper position variables to subtract.

$$\text{if } \eta_S = 1: \quad l \leq x_{R_{i+1}} - y_{R_i} \leq u \tag{5.12}$$

$$\text{if } \eta_S = 0: \quad l \leq -(x_{R_{i+1}} - y_{R_i}) \leq u \tag{5.13}$$

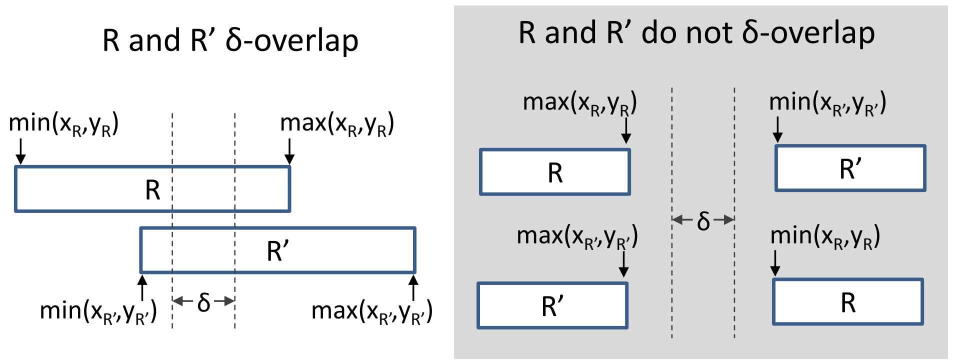

Figure 5.2: **δ-Overlapping vs. non-δ-overlapping strobes in a strobe layout.** Here, the subread orientations are ignored as the minimum or maximum value for each coordinate is selected. If subreads R and R' δ-overlap, then both $(\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1)$ and $(\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1)$ are greater than or equal to δ. If they do not δ-overlap, then either $(\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1)$ or $(\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1)$ is less than δ.

Overlapping Constraints These constraints ensure that $\beta_{R,R'} = 1$ if and only if R and R' δ-overlap in the strobe layout. The constraints subtract the minimum overlap length δ from the inner coordinates of R and R' and ensure that this is non-negative if $\beta_{R,R'}$ is one (Figure 5.2).

$$\text{if } \beta_{R,R'} = 1: \quad \begin{cases} \delta \geq (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \\ \delta \geq (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \end{cases} \tag{5.14}$$

$$\text{if } \beta_{R,R'} = 0: \quad \begin{cases} \delta < (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) & \text{if } \max(x_R, y_R) < \min(x_{R'}, y_{R'}) \\ \delta < (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) & \text{if } \max(x_{R'}, y_{R'}) < \min(x_R, y_R) \end{cases} \tag{5.15}$$

In addition to $\beta_{R,R'}$, we introduce another binary variable $\alpha_{R,R'}$ that is 1 if R is to the right of R' (and $\max(x_{R'}, y_{R'}) < \min(x_R, y_R)$) and 0 if R is to the left of R' (and $\max(x_R, y_R) < \min(x_{R'}, y_{R'})$). Then, the inequalities are written as

$$\text{if } \beta_{R,R'} = 1: \quad \begin{cases} \delta \geq (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \\ \delta \geq (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \end{cases} \tag{5.16}$$

$$\text{if } \beta_{R,R'} = 0 \text{ and } \alpha_{R,R'} = 0: \quad \delta < (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \tag{5.17}$$

$$\text{if } \beta_{R,R'} = 0 \text{ and } \alpha_{R,R'} = 1: \quad \delta < (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \tag{5.18}$$

Configuration Constraints Note that the overlapping constraints above do not consider subread orientation for subread R. Thus, two subreads R and R' might δ-overlap, but not in the appropriate configuration determined by the pairwise alignments. For example, R and R' might δ-overlap in the layout, while the pairwise alignment of R and R' involve R and the reverse complement of R', R'^c. In this case, we do *not* want R and R' to δ-overlap, only R and R'^c. Further, if R and R' do δ-overlap, they should overlap in the same approximate configuration as the pairwise alignment of R and R'.

Thus, we further restrict the placement of the x and y variables for δ-overlapping subreads R and R' by requiring that $(x_{R'} - x_R)$ is at most ω bases away from $\mathcal{H}_1(R, R')$ and $(y_{R'} - y_R)$ is at most ω bases away from $\mathcal{H}_2(R, R')$. The user-defined parameter ω is necessary in practice when laying out a set of subreads that all pairwise align with each other, and substitution and indel errors result in optimal pairwise alignments with slightly different configurations. While ω accounts for the potential shift in the overlap configurations, we must also include variability due to the indels, $\varepsilon(\cdot)$.

First, we define the overlap configurations $\hat{\mathcal{H}}_1(R, R')$ and $\hat{\mathcal{H}}_2(R, R')$ in terms of the position variables x and y. As with with alignment configurations $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$, we compute the overlap configurations $\hat{\mathcal{H}}_1(R, R')$ and $\hat{\mathcal{H}}_2(R, R')$ such that the first subread R is in the forward orientation. We use the orientation variable η_S to determine $\hat{\mathcal{H}}_1(R, R')$ and $\hat{\mathcal{H}}_2(R, R')$ (Figure 5.3).

$$\text{if } \eta_S = 1: \quad \begin{cases} \hat{\mathcal{H}}_1(R, R') = x_{R'} - x_R \\ \hat{\mathcal{H}}_2(R, R') = y_{R'} - y_R \end{cases} \tag{5.19}$$

$$\text{if } \eta_S = 0: \quad \begin{cases} \hat{\mathcal{H}}_1(R, R') = -(x_{R'} - x_R) \\ \hat{\mathcal{H}}_2(R, R') = -(y_{R'} - y_R) \end{cases} \tag{5.20}$$

We can now compare the overlap configurations $\hat{\mathcal{H}}_1(R, R')$ and $\hat{\mathcal{H}}_2(R, R')$ to the alignment configurations $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ for subreads R and R',

$$\text{if } \beta_{R,R'} = 1: \quad \begin{cases} -\omega - \varepsilon(|\mathcal{H}_1(R, R')|) \leq \hat{\mathcal{H}}_1(R, R') - \mathcal{H}_1(R, R') \leq \omega + \varepsilon(|\mathcal{H}_1(R, R')|) \\ -\omega - \varepsilon(|\mathcal{H}_2(R, R')|) \leq \hat{\mathcal{H}}_2(R, R') - \mathcal{H}_2(R, R') \leq \omega + \varepsilon(|\mathcal{H}_2(R, R')|) \end{cases} \tag{5.21}$$

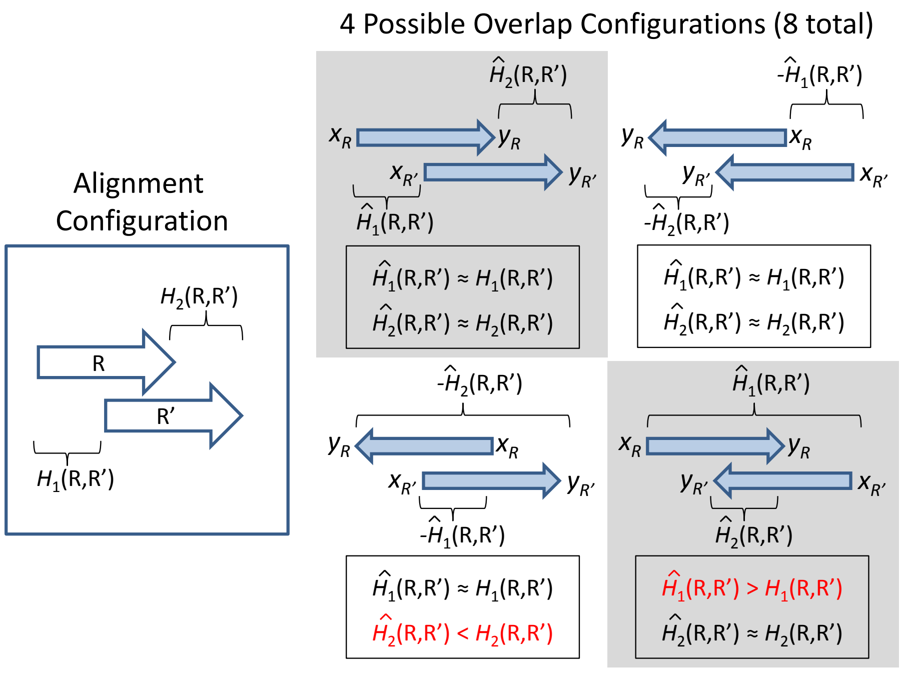
4 Possible Overlap Configurations (8 total)
Alignment Configuration
R
R'
H_1(R,R')
H_2(R,R')
x_R
y_R
$x_{R'}$
$y_{R'}$
$\hat{H}_1$(R,R')
$\hat{H}_2$(R,R')
-$\hat{H}_1$(R,R')
-$\hat{H}_2$(R,R')
$\hat{H}_1$(R,R') ≈ H_1(R,R')
$\hat{H}_2$(R,R') ≈ H_2(R,R')
$\hat{H}_2$(R,R') < H_2(R,R')
$\hat{H}_1$(R,R') > H_1(R,R')

Figure 5.3: **Four of eight possible overlap configurations.** The inequalities in red are instances where $\hat{\mathcal{H}}_1(R, R')$ is not near $\mathcal{H}_1(R, R')$ or $\hat{\mathcal{H}}_2(R, R')$ is not near $\mathcal{H}_2(R, R')$; thus, R and R' cannot δ-overlap according to these configurations. The other four overlap configurations include subread R being to the right of R'; in those configurations, none of $\hat{\mathcal{H}}_1(R, R')$ or $\hat{\mathcal{H}}_2(R, R')$ are near $\mathcal{H}_1(R, R')$ or $\mathcal{H}_2(R, R')$, respectively.

These inequalities ensure that there will be no strobe layouts in which R and R' δ-overlap but are not near the proper alignment configuration.

The Complete Set of Constraints. Putting it all together, the Strobe Assembly Problem corresponds to satisfying the following set of (not necessary linear) constraints with boolean conditionals:

Maximize:

$$\max_{\mathbf{x},\mathbf{y}} Q(\mathcal{L}_{\mathbf{S}})$$

Subject to

$$\forall S, S' \in \mathbf{S}, \forall R \in S, \forall R' \in S' :$$

Position and Advance Constraints:

$$\text{if } \eta_S = 1 : \quad \begin{cases} 0 < y_R - x_R \\ |R| - \varepsilon(|R|) \leq (y_R - x_R + 1) \leq |R| + \varepsilon(|R|) \\ l \leq x_{R_{i+1}} - y_{R_i} \leq u \end{cases} \tag{5.22}$$

$$\text{if } \eta_S = 0 : \quad \begin{cases} 0 < -(y_R - x_R) \\ |R| - \varepsilon(|R|) \leq (-(y_R - x_R) + 1) \leq |R| + \varepsilon(|R|) \\ l \leq -(x_{R_{i+1}} - y_{R_i}) \leq u \end{cases} \tag{5.23}$$

Overlapping Constraints:

$$\text{if } \beta_{R,R'} = 1 : \quad \begin{cases} \delta \geq (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \\ \delta \geq (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \end{cases} \tag{5.24}$$

$$\text{if } \beta_{R,R'} = 0 \text{ and } \alpha_{R,R'} = 0 : \quad \delta < (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \tag{5.25}$$

$$\text{if } \beta_{R,R'} = 0 \text{ and } \alpha_{R,R'} = 1 : \quad \delta < (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \tag{5.26}$$

Configuration Constraints:

$$\text{if } \beta_{R,R'} = 1 : \quad \begin{cases} -\omega - \varepsilon(|\mathcal{H}_1(R,R')|) \leq \hat{\mathcal{H}}_1(R,R') - \mathcal{H}_1(R,R') \leq \omega + \varepsilon(|\mathcal{H}_1(R,R')|) \\ -\omega - \varepsilon(|\mathcal{H}_2(R,R')|) \leq \hat{\mathcal{H}}_2(R,R') - \mathcal{H}_2(R,R') \leq \omega + \varepsilon(|\mathcal{H}_2(R,R')|) \end{cases} \tag{5.27}$$

where $C > 0$ is a sufficiently large integer, and ω is the number of bases allowed to deviate from the configuration determined by $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$.

We first verify that a selection of position variables $\mathbf{x}$ and $\mathbf{y}$ that maximize $Q(\mathcal{L}_\mathbf{S})$ with respect to the above constraints is a strobe layout with an optimal layout score.

Theorem 5.2.1. *A selection of position variables* $\mathbf{x}$ *and* $\mathbf{y}$ *that maximize* $Q(\mathcal{L}_\mathbf{S})$ *with respect to constraints* (5.22) - (5.27) *is a strobe layout* $\mathcal{L}^*_\mathbf{S}$ *with a maximum score* $Q(\mathcal{L}^*_\mathbf{S})$.

Proof. We first prove that a selection of $\mathbf{x}$ and $\mathbf{y}$ that respects constraints (5.22)-(5.27) is a strobe layout $\mathcal{L}_\mathbf{S}$. A strobe layout requires that, for each strobe,

1. The coordinates (x_R, y_R) for each subread $R \in \mathbf{R}$ indicate at most $\varepsilon(|R|)$ insertions or deletions in R (Equation (5.2)). The second constraint in (5.22) ensures this if R is in the forward orientation and the second constraint in (5.23) ensures this if R is in the reverse orientation.

2. All subreads must be in the same orientation (Equation (5.3)). The first constraint in (5.22) ensures this if strobe S is in the forward orientation and and the first constraint in (5.23) ensure this if strobe S is in the reverse orientation.

3. The advance lengths are bounded by l and u (Equation (5.4)). The third constraint in (5.22) ensures this if strobe S is in the forward orientation and and the third constraint in (5.23) ensure this if strobe S is in the reverse orientation.

Second, we prove that the solution to the ILP is maximal in terms of $Q(\mathcal{L}_\mathbf{S})$. To maximize the objective function properly, the binary overlap variables $\beta_{R,R'}$ must be 1 if and only if subreads R and R' δ-overlap in the solution *and* if that δ-overlap represents the pairwise alignment of R and R'. The constraint in (5.24) ensure that $\beta_{R,R'} = 1$ if R and R' δ-overlap in the layout. Since $\alpha_{R,R'}$ must be zero or one, then one of (5.25) or (5.26) ensures that $\beta_{R,R'} = 0$ if R and R' do not δ-overlap. Finally, the configuration constraint (5.27) ensures that the δ-overlapping subreads reflect the pairwise alignment configuration up to ω bases. □

From Boolean Conditions to Linear Constraints. To convert (5.22) - (5.27) into linear constraints, we employ a trick that allows "if/else" boolean conditions to become a set of linear constraints.

Lemma 5.2.2. *Given a binary variable* B *and two linear inequalities* $X \sim 0$ *and* $Y \dot{\sim} 0$*, where* $\sim$ *and* $\dot{\sim}$ *are one of* $\{<, \leq, >, \geq\}$*, a boolean condition of the form*

$$\textit{if } B = 1 : \quad X \sim 0 \tag{5.28}$$

$$\textit{if } B = 0 : \quad Y \dot{\sim} 0 \tag{5.29}$$

can be transformed into a set of two linear constraints.

Proof. The main idea is to list X and Y and evaluate one or the other depending on the value of B. To do so, we add a term to each of X and Y that makes the resulting inequality trivial for a certain value of B.

We first introduce a sufficiently large constant C to each constraint such that they are both automatically satisfied,

$$\text{if } B = 1: \quad \begin{cases} X \sim C & \text{if } \sim \in \{<, \leq\} \\ X \sim -C & \text{if } \sim \in \{>, \geq\} \end{cases} \tag{5.30}$$

$$\text{if } B = 0: \quad \begin{cases} Y \dot{\sim} C & \text{if } \dot{\sim} \in \{<, \leq\} \\ Y \dot{\sim} -C & \text{if } \dot{\sim} \in \{>, \geq\} \end{cases}. \tag{5.31}$$

Now, we use B to set a coefficient of C to be zero or one. $X \sim 0$ should be satisfied when $B = 1$ and trivial if $B = 0$; thus, we set the coefficient of C to be $(1 - B)$ in the first inequality. Alternatively, $Y \dot{\sim} 0$ should be satisfied when $B = 0$ and trivial if $B = 1$; thus, we set the coefficient of C to be B in the second inequality. For example, for $X > 0$ and $Y < 0$ the boolean condition is equivalent to

$$X > -(1 - B)C \tag{5.32}$$

$$Y < BC. \tag{5.33}$$

Such constraints are specified for general $X \sim 0$ and $Y \dot{\sim} 0$, with $\sim, \dot{\sim} \in \{<\leq, >, \geq\}$. □

Note that we may use this lemma by recursively applying it to inequalities whose non-linear terms may be described as boolean conditions. Using this trick, we can now write the integer linear program.

Lemma 5.2.3. *Maximizing $Q(\mathcal{L}_\mathbf{S})$ with respect to constraints* (5.22) - (5.27) *over position variables* $\mathbf{x}$ *and* $\mathbf{y}$ *can be written as an equivalent integer linear program.*

Proof. First, observe that $Q(\mathcal{L}_\mathbf{S})$ is a linear objective function; thus, it is a proper objective function for an ILP. We now describe how each of the constraints in (5.22) - (5.27) may be transformed into a set of linear constraints by recursively applying Lemma 5.2.2.

1. **Position and Advance Constraints** (5.22) **and** (5.23)**.** With the use of a large constant C, the boolean

conditions in the position and advance constraints are transformed into linear constraints,

$$-(1-\eta_S)C < y_R - x_R \tag{5.34}$$

$$-(1-\eta_S)C + |R| - \varepsilon(|R|) \leq (y_R - x_R + 1) \leq |R| + \varepsilon(|R|) + (1-\eta_S)C \tag{5.35}$$

$$-(1-\eta_S)C + l \leq x_{R_{i+1}} - y_{R_i} \leq u + (1-\eta_S)C \tag{5.36}$$

$$-\eta_S C < -(y_R - x_R) \tag{5.37}$$

$$-\eta_S C + |R| - \varepsilon(|R|) \leq (-(y_R - x_R) + 1) \leq |R| + \varepsilon(|R|) + \eta_S C \tag{5.38}$$

$$-\eta_S C + l \leq -(x_{R_{i+1}} - y_{R_i}) \leq u + \eta_S C \tag{5.39}$$

2. **Overlapping constraints** (5.24) **-** (5.26)**.** The overlapping constraints incorporate both $\beta_{R,R'}$ and $\alpha'_{R,R}$ into the coefficient of C:

$$(1-\beta_{R,R'})C + \delta \geq (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \tag{5.40}$$

$$(1-\beta_{R,R'})C + \delta \geq (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \tag{5.41}$$

$$-(\beta_{R,R'} + \alpha_{R,R'})C + \delta < (\max(x_R, y_R) - \min(x_{R'}, y_{R'}) + 1) \tag{5.42}$$

$$-(\beta_{R,R'} + (1-\alpha_{R,R'}))C + \delta < (\max(x_{R'}, y_{R'}) - \min(x_R, y_R) + 1) \tag{5.43}$$

Note that these constraints are not linear due to the $\min$ and $\max$ terms. However, the $\min$ and $\max$ terms can be written as boolean conditions,

$$\text{if } \eta_S = 1: \quad \begin{cases} \min(x_R, y_R) = x_R \\ \max(x_R, y_R) = y_R \end{cases} \tag{5.44}$$

$$\text{if } \eta_S = 0: \quad \begin{cases} \min(x_R, y_R) = y_R \\ \max(x_R, y_R) = x_R \end{cases}. \tag{5.45}$$

Thus, these constraints with $\min$ and $\max$ terms may be transformed to linear constraints. For example, (5.40) becomes four linear constraints,

$$\left((1-\beta_{R,R'}) + (1-\eta_S) + (1-\eta_{S'})\right) C + \delta \geq y_R - x_{R'} + 1 \tag{5.46}$$

$$\left((1-\beta_{R,R'}) + \eta_S + (1-\eta_{S'})\right) C + \delta \geq x_R - x_{R'} + 1 \tag{5.47}$$

$$\left((1-\beta_{R,R'}) + (1-\eta_S) + \eta_{S'}\right) C + \delta \geq y_R - y_{R'} + 1 \tag{5.48}$$

$$\left((1-\beta_{R,R'}) + \eta_S + \eta_{S'}\right) C + \delta \geq x_R - y_{R'} + 1. \tag{5.49}$$

3. **Configuration Constraints** (5.27)**.** The configuration constraints may be written as the following.

We slightly abuse notation for readability by defining $e_1 = \omega + \varepsilon(|\mathcal{H}_1(R,R')|)$ and $e_2 = \omega + \varepsilon(|\mathcal{H}_2(R,R')|)$ to be the number of bases that the overlap configuration may be off by the alignment configuration.

$$-(1-\beta_{R,R'})C - e_1 \leq \hat{\mathcal{H}}_1(R,R') - \mathcal{H}_1(R,R') \leq e_1 + (1-\beta_{R,R'})C \tag{5.50}$$

$$-(1-\beta_{R,R'})C - e_2 \leq \hat{\mathcal{H}}_2(R,R') - \mathcal{H}_2(R,R') \leq e_2 + (1-\beta_{R,R'})C \tag{5.51}$$

However, these are not linear yet due to the calculation of $\hat{\mathcal{H}}_1(R,R')$ and $\hat{\mathcal{H}}_2(R,R')$ in (5.19) and (5.20). Thus, we apply the lemma again to produce linear constraints:

$$-\big((1-\beta_{R,R'}) + (1-\eta_S)\big)C - e_1 \leq (x_{R'} - x_R) - \mathcal{H}_1(R,R') \leq e_1 + \big((1-\beta_{R,R'}) + (1-\eta_S)\big)C \tag{5.52}$$

$$-\big((1-\beta_{R,R'}) + (1-\eta_S)\big)C - e_2 \leq (y_{R'} - y_R) - \mathcal{H}_2(R,R') \leq e_2 + \big((1-\beta_{R,R'}) + (1-\eta_S)\big)C \tag{5.53}$$

$$-\big((1-\beta_{R,R'}) + \eta_S\big)C - e_1 \leq -(x_{R'} - x_R) - \mathcal{H}_1(R,R') \leq e_1 + \big((1-\beta_{R,R'}) + \eta_S\big)C \tag{5.54}$$

$$-\big((1-\beta_{R,R'}) + \eta_S\big)C - e_2 \leq -(y_{R'} - y_R) - \mathcal{H}_2(R,R') \leq e_2 + \big((1-\beta_{R,R'}) + \eta_S\big)C \tag{5.55}$$

Since constraints (5.22)-(5.27) are all transformed into linear constraints, then these provide proper constraints for an integer linear program. □

We note that some of these linear inequalities may be simplified; in particular, the configuration constraints may be defined using only half of the number of constraints described above, as η_S may be used to determine whether to add or subtract $\mathcal{H}_1(R,R')$ and $\mathcal{H}_2(R,R')$. However, this does not reduce the number of binary variables, so for ease of exposition we leave the constraints as described. The complete ILP is written in §A

Program Analysis For a set **S** of t-strobes, where $|\mathbf{S}| = n$, there are $2(nt)^2$ variables due to the β and α variables required for the overlapping and configuration constraints. However, we do not compute alignments between subreads from the same strobe; thus, there are $\binom{t}{2}$ variables that are fixed. There are $(2nt)$ position variables for the x's and the y's. With another n variables for the orientation vector η, there are $O(2(nt)^2 - n\binom{t}{2} + 2nt + n)$ variables, which is quadratic in the number of subreads.

Remark 5.2.4. *In the single-stranded case, where there is no reverse complement strand, $\eta_i = 1$ for all strobes and the orientation vector is not necessary. Thus, the number of variables is reduced by n.*

Remark 5.2.5. *If there are no indels in the subreads, we have the following additional equalities:*

$$y_R = (x_R + |R| - 1) \textit{ or } (x_r - (|R| - 1))$$

$$\mathcal{H}_2(R, R') = \begin{cases} \mathcal{H}_1(R, R') & \textit{if } R \textit{ aligns to } R' \\ |R| + |R'| + \mathcal{H}_1(R, R') & \textit{if } R \textit{ aligns to } R'^c \end{cases}$$

Thus, the y variables can be written in terms of x variables and η variables, and the number of variables is reduced by nt.

5.2.3 The ILP Assembler

We have shown that a solution to the integer linear program defined above will produce a strobe layout. However, the number of variables in this program increases rapidly with the number of strobes. In particular, while the number of position and orientation variables scales linearly with the number of strobes, the number of α and β variables increases quadratically. Thus, an exact solution to the ILP can only be obtained for only a small number of strobes. We employ a number strategies to reduce the number of variables in the ILP.

Banning Overlaps. Based on pairwise alignments of subreads, we know that the majority of subread pairs are highly unlikely to δ-overlap in the true assembly (when $o(R, R')$, for example). Thus, we set the corresponding β variables to zero and do not consider them in the ILP. Each *banned* overlap reduces the number of variables by one.

Forcing Overlaps. There may be pairwise subread alignments that we want to ensure are in the solution. Thus, we set the corresponding β variables to one. Additionally, since the α variables are used only for non-overlapping conditions then we arbitrarily set α. Each *forced* overlap reduces the number of variables by two.

Incorporating Relative Ordering of Subreads. Forcing the overlap of two subreads provides additional information about the other subreads in the relevant strobes. We define a subread R_{i_a} to *contain* another subread R_{j_b} if the forced overlap between R_{i_a} and R_{j_b} is such that $o(R_{i_a}, R_{j_b}) = r_{j_b}$.

1. **Relative Subread Ordering.** Suppose we have four subreads a,b,c, and d from different strobes, and a contains b and c contains d. If a and c do not overlap, then the relative ordering of b and d are the same as the relative ordering of a and c. That is, $\alpha_{a,c} = \alpha_{b,d}$. If a and c do overlap, then the choice of α variables is arbitrary so they can still be equal. Specifying relative subread ordering reduces the number of α variables if there are a few subreads that contain many other subreads.

2. **Ignoring Impossible Alignments.** Now suppose we have three subreads R_{i_a},$R_{i_{a+1}}$, and R_{j_b}, where R_{j_b} contains $R_{i_{a+1}}$ and R_{j_b} aligns to R_{i_a} with some $\mathcal{H}_1(R, R')$ and some $\mathcal{H}_2(R, R')$. We can compute

the advance length between the contained subread and R_{i_a} if the alignment to R_{j_b} and R_{i_a} is used in the solution. If the advance length is now within the bounds l and u, then the ILP solution will never choose this alignment for R_{j_b} and R_{i_a}. Thus, we may ban this alignment or find a suboptimal alignment of R_{j_b} and R_{i_a} that respects the advance lengths.

Exploiting the solution structure to reduce the number of variables for the ILP using the strategies above results in obtaining an exact solution to the ILP for a few dozen strobes. While this prevents a large number of strobes to be assembled at the same time, we effectively assemble larger fragments by greedily assembling a small number of strobes at a time, aggregating the assembled strobes in an iterative fashion by treating the layout from one iteration as input strobes for the next iteration. This iterative method allows us to force overlaps from previous iterations in the current iteration, which greatly reduces the number of variables (Figure 5.5).

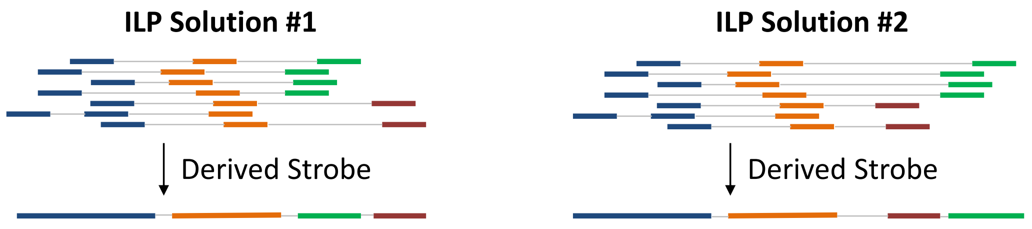

Figure 5.4: **A Derived Strobe from a layout of 3-strobes.** The layout of 3-strobes produces a set of contigs, which represent derived subreads that are linked by pairing information to produce a derived strobe. Different layouts might produce different relative contig orderings; thus, while bounds are determined for derived strobes, their absolute order cannot be determined.

Scaffolds as Derived Strobes. The iterative method hinges on the following observation. An ILP solution provides a strobe layout consisting of sets of contigs organized into scaffolds. Suppose a scaffold consists of k contigs. If we derive consensus sequences for each contig, we end up with k strings over the alphabet Σ. Additionally, we can derive bounds on the distance between contigs in the scaffold. Thus, a scaffold is similar to the definition of a t-strobe with different subread lengths and distance bounds on the advances between the t subreads (Figure 5.4). Note that a scaffold is not quite the same as a strobe because of the partial ordering of the contigs. We call strobes that are determined by scaffolds *derived strobes* to differentiate them from the original input strobes, and we call the contigs *derived subreads*. Note that we still use the term *contigs* to indicate the corresponding set of subreads, rather than the consensus sequence.

The notion of derived strobes reduces the number of variables in the ILP because every subread that appears in a contig is a *forced* overlap. Additionally, we are now able to input a set of initial contigs into the ILP that are easy to assemble, and let the ILP make the hard decisions for difficult regions.

There are six main components of the ILP Assembler: algorithm initialization, sampling strobes, determining pairwise alignments, running the ILP, converting the ILP solution to an assembly, and algorithm

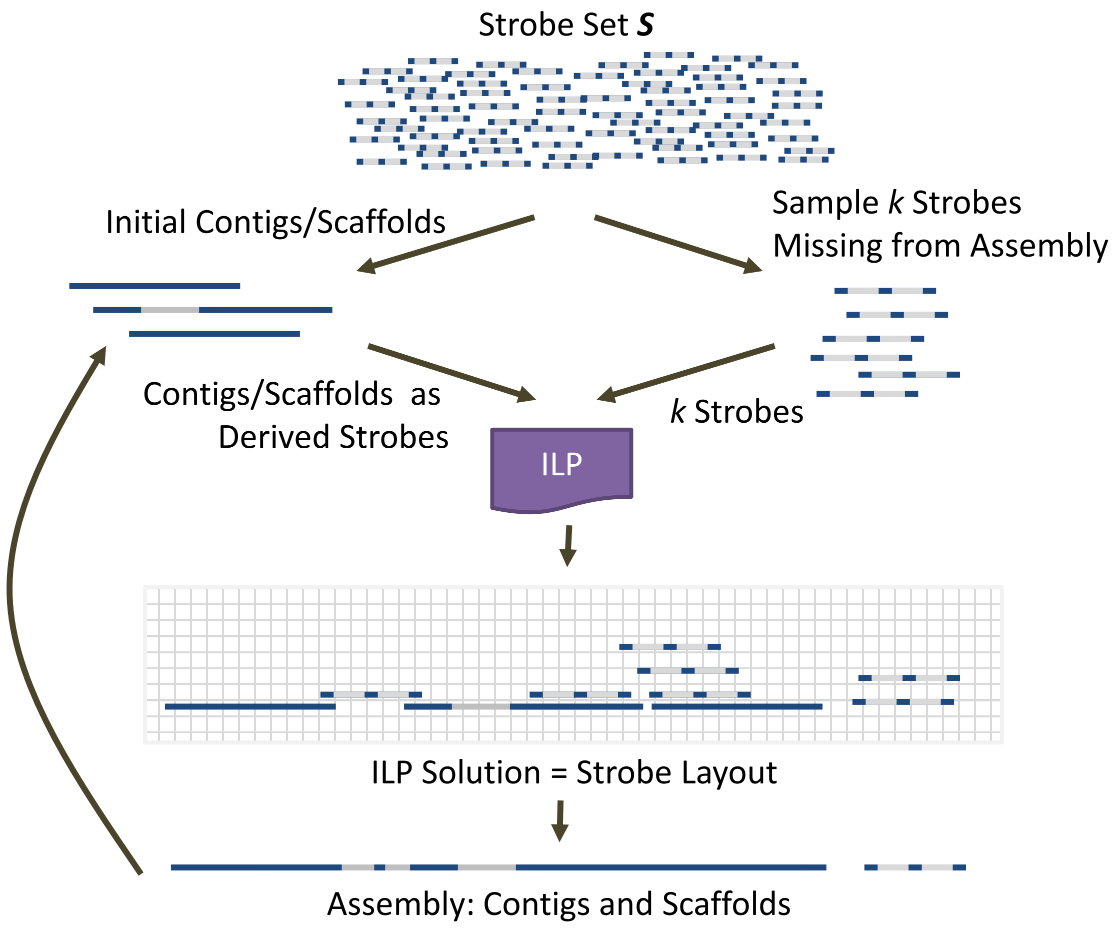
Strobe Set ***S***
Initial Contigs/Scaffolds
Sample *k* Strobes Missing from Assembly
Contigs/Scaffolds as Derived Strobes
k Strobes
ILP
ILP Solution = Strobe Layout
Assembly: Contigs and Scaffolds

Figure 5.5: **Overview of the Iterative ILP Assembler.** A set of initial contigs are established from the set of strobes S. These contigs are formulated as derived strobes, combined with k strobes sampled from the remainder of the set, and run through the ILP. The resulting strobe layout defines a set of new contigs and scaffolds that are then passed as derived strobes in the next iteration. The method terminates when no ILP solution is found or all the strobes are in the assembly.

termination. We run the ILP using the CPLEX ILP solver, and we detail the other components below.

Algorithm Initialization. We begin with a set of initial contigs and scaffolds for easy-to-assemble regions; we find these by constructing the fragment assembly string graph [96] from the top 90% pairwise alignments that δ-overlap and selecting the longest edges from this graph. We then determine the derived strobes from these scaffolds and combine them with a set of k strobes sampled from the strobe set $\mathbf{S}$.

Sampling Strobes. We sample k strobes in the following way. First, we say that a subread is a *floating subread* if it is not in the set of contigs and a *fixed subread* otherwise. We sample $k/2$ strobes with at least one floating subread and at least one fixed subread (called *anchored strobes*), and we sample $k/2$ strobes with no floating subreads (called *unanchored* strobes). Anchored strobes are constrained due to forced overlaps of the fixed subreads to the corresponding derived subreads. Unanchored strobes, however, are only constrained by pairing information and the subread overlaps. If we sampled unanchored strobes uniformly, they might overlap with very few other strobes. If a strobe is under-constrained, we are in danger of placing it in the incorrect position. Thus, we employ an sampling technique to avoid this issue (Figure 5.6). We arbitrarily pick a derived subread and sample $k/2$ unanchored strobes where at least one floating subread has a good alignment score with the derived subread. This localized sampling technique reduces the chance that we misplace a strobe due to poor alignment information.

Determining Pairwise Overlaps. To run the ILP on the derived strobes and the k sampled strobes, we must first determine all pairwise overlaps in the set. We ban all overlaps between derived subreads because we assume that contigs do not overlap in the assembly. We also ban any overlap between subreads with a smith-waterman alignment score of 0 and any alignment that is not a δ-overlap. We force overlaps between fixed subreads and their corresponding derived subreads, and we ban overlaps between fixed subreads and the other derived subreads. To determine the overlap of a floating subread R_{i_a} and a derived subread R_{j_b}, we find the subread within the corresponding contig that has the best alignment score and calculate the alignment offsets $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$. If this alignment is incorrect or there is a misassembly within the contig, $\mathcal{H}_1(R, R')$ and $\mathcal{H}_2(R, R')$ might represent an impossible alignment. If this is the case, we choose the subread with the next best alignment, and so on. With this alignment information, we run the ILP on the derived strobes and the k sampled strobes to produce a strobe layout $\mathcal{L}_{\mathbf{S}}$.

Converting the ILP Solution to an Assembly. Once we find $\mathcal{L}_{\mathbf{S}}$, we collapse the strobes and derived strobes into a new set of contigs and scaffolds, where a contig contains subreads that δ-overlap in the strobe layout as well as any original subreads from the derived strobes that δ-overlap. However, we must do this conversion carefully, because the strobe layout provided by the solution is only correct up to the relaxation parameter ω. In other words, if shifting one subread by a few bases satisfies all constraints and does not change the optimal layout score, then the ILP solver will arbitrarily pick one of these solutions. Thus,

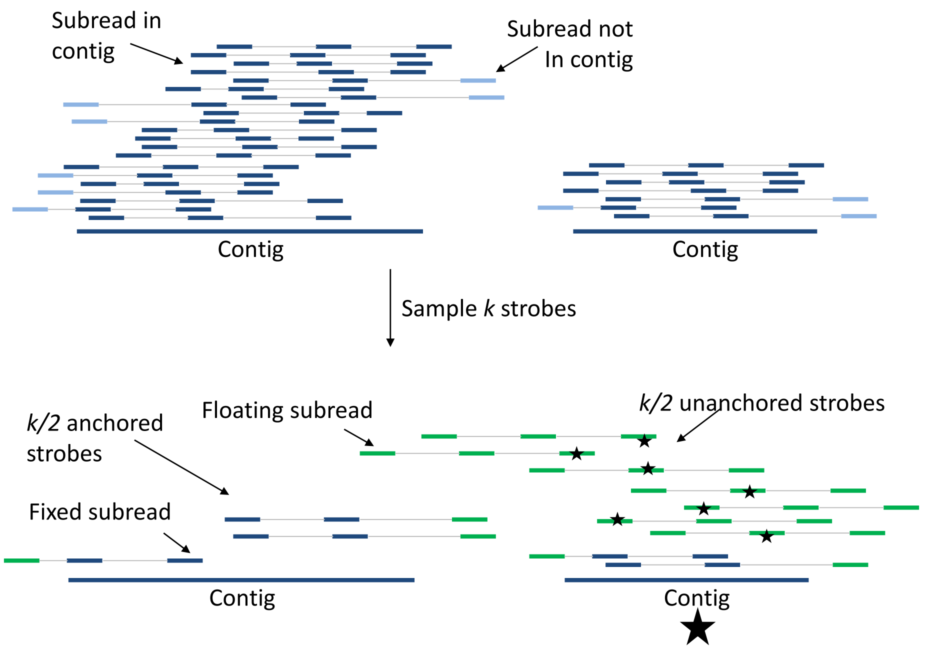

Figure 5.6: **Sampling technique used in the ILP Assembler.** At a particular iteration in the assembler, there is a set of contigs that contain *anchored* strobes; strobes where at least one subread is part of the contig and at least one subread is not part of any contig (called a *floating*) subread. When we sample k strobes, $k/2$ of them are anchored strobes (blue) and $k/2$ of them are unanchored strobes (green). Starred subreads denote good alignments with the starred contig. Good alignments (determined from the ILP run) are then merged into the contig for the subsequent iteration.

we refine the strobe layout by going back to the subread sequences (for strobes) and consensus sequences (for derived strobes) and re-aligning all δ-overlapping pairs according to $\mathcal{L}_\mathbf{S}$. We do this in linear time in a greedy fashion by scanning the layout from left to right and aligning pairs as we reach them; if the subreads or derived subreads are very long we only align the first 500 bases. After we align we recompute the position coordinates, which may shift up to ω bases. Once we have refined the layout, constructing the consensus sequence from a substitution-only error model is trivial: each base is chosen by majority vote. Once the consensus sequences for the contigs are established, scaffolds become the derived strobes in the next iteration of the algorithm, k strobes are sampled once again, and the ILP is run.

Algorithm Termination. If there are no misassemblies, the algorithm terminates when all strobes are incorporated into the assembly. However, the algorithm may terminate early if a misassembly is detected and the method is unable to recover. There are many ways that misassemblies manifest in this method.

1. A subread in $\mathcal{L}_\mathbf{S}$ does not align well to other subreads that δ-overlap according to $\mathcal{L}_\mathbf{S}$. This may occur because the sampled set of strobes do not include any real overlaps to this subread and thus an incorrect alignment with a poor score is chosen, and is partly corrected for with importance sampling described above. Our method can detect misassemblies such as this in the layout refinement stage: if the alignment of two subreads shifts the layout by more that ω bases, we assume that one of the subreads is misplaced and the smaller of the subreads is removed. The removed subreads are placed back in the set of strobes to sample, and will be chosen in a future iteration.

2. A strobe in $\mathcal{L}_\mathbf{S}$ results in a scaffold whose scaffold graph is *not* a DAG. This means that there's no possible linear order of contigs in the scaffold. Our method detects this and removes the entire scaffold. If parts of the affected scaffold included initial contigs, then these are added again as derived strobes. All other strobes are placed back in the set of strobes to sample, and will be chosen in a future iteration.

3. Other misassemblies result in an infeasible ILP and the ILP solver terminates. The last set of contigs and scaffolds are returned as the final assembly.

The number of iterations before the algorithm terminates varies drastically by the size of the input contigs, the number of sampled strobes k, and the alignment thresholds t_O and t_N.

5.3 Results

We apply the ILP Assembler to simulated data from four different bacterial artificial chromosomes (BACs) that have been hard to assemble with current technologies [27]: they are described in Table 5.1.

Dataset	GenBank ID	Contig length	# of Repeats	Repeat % of Sequence	Prevalent Repeat Families
B0	AF045449	32,613	36	17.90%	Alu,L2,Mir,Mlt
B1	AF045450	40,205	54	62.60%	Alu,Herv,Mlt
B2	AF129076	42,051	39	32.20%	Alu,L1,Mer,Mlt
B3	AF015722	47,162	71	15.60%	Alu,L1,Mer,The1b

Table 5.1: **Hard-to-assemble BACs Selected for Strobe Sequencing Simulations.**

We first demonstrate that the ILP Assembler accurately assembles these BACs with sequencing error as high as 10%. We then show that our method outperforms current assemblers, even when they are provided with the pairing information from strobes and optimal subread alignments.

5.3.1 ILP Assembler Results

Data Simulation. For each BAC sequencing, we simulated 3Kb strobes on the forward strand consisting of three 200bp subreads and two 1200bp advances at 30X (B0-B3) or 15X (B4-B6) sequence coverage. We then flipped the strobe orientation with probability 0.5, which included determining the reverse complements of the sequences and reversing the subread order. Finally, we introduced 1%, 5%, and 10% substitution error into the sequences, producing three different datasets.

ILP Assembler Parameters. We computed the end-space free alignments for all pairs of subreads that were not from the same strobe using a score of $+1$ for a match and -3 for a mismatch (10% error) or -4 for a mismatch (5% and 1% error). The indel penalty is 1000, ensuring that no alignment contains insertions or deletions. We immediately discard any alignments that do not δ-overlap, where $\delta = 30$.

To determine initial contigs, we constructed the fragment assembly string graph for the top 90% of pairwise alignments, and we simplified the graph by removing nodes v with a single incoming edge (u, v) and a single outgoing edge (v, x) and merging the overlaps from (u, v) and (v, x) into a single edge (u, x). We select edges in the graph with more than 200 subreads involved in the alignments; we take care that we don't select two edges that represent the same region of the genome (both the forward and reverse complement). We then convert the edges to layouts, and trim subreads within 200bp of the ends of the layouts.

We run the ILP Assembler using CPLEX with $k = 20$ and $\omega = 10$ and alignment thresholds of $t_N = 0$ and $t_O = 1$. We observed that while some iterations took hours for CPLEX to find an optimal solution, most of the iterations took less than two minutes. Thus, we set a time limit on CPLEX to return after 5 minutes, and if a solution is not found within this time, we resample k and re-run CPLEX. If CPLEX returns a suboptimal solution, we found that the layout refinement corrected any potential misassemblies produced by the suboptimal layout; thus we use suboptimal solutions if they are found within 5 minutes. Results are in Table 5.2.

	Contigs from String Graph					ILP Assembler				
	Contigs		Scaffolds		%	Contigs		Scaffolds		%
Dataset	#	N50	#	N50	Assembled	#	N50	#	N50	Assembled
B0 1%	4	9285	1	29659	91%	5	9285	1	30177	93%
B0 5%	5	14521	1	27103	83%	7	10609	1	29589	90%
B0 10%	4	14387	1	25594	80%	5	14423	1	28243	82%
B1 1%	9	4730	1	30250	76%	1	40188	1	40188	100%
B1 5%	8	4893	1	19882	76%	1	40188	1	40188	100%
B1 10%	7	3963	1	18896	67%	1	40188	1	40188	100%
B2 1%	3	32676	1	40876	96%	1	42034	1	42034	100%
B2 5%	6	13587	1	40062	93%	6	25746	1	44506	100%
B2 10%	4	32683	1	40705	95%	1	42034	1	42034	100%
B3 1%	8	9653	1	34797	83%	1	41061	1	41061	100%
B3 5%	7	9653	1	34630	82%	1	41091	1	41091	100%
B3 10%	9	3927	2	19463	66%	2	31838	1	44003	100%

Table 5.2: **Initial Contigs and ILP Assembly Results.** The initial contigs (Left) are constructed using the string graph representation [96], and are used as the input to the ILP algorithm. The ILP algorithm (Right) further completes the assemblies by adding unassembled subreads and merging contigs.

5.3.2 Comparison to Existing Assemblers

We compared our method to two assemblers: Velvet [158] and Bambus [140]. For both algorithms, we input all the pairing information from the strobe sequencing dataset in the following way. For each 3-strobe, we produced three mate-pairs that are simulated from two different insert libraries (Figure 5.7). This ensured that the same amount of information was given to the Velvet and Bambus that was available in the ILP Assembler.

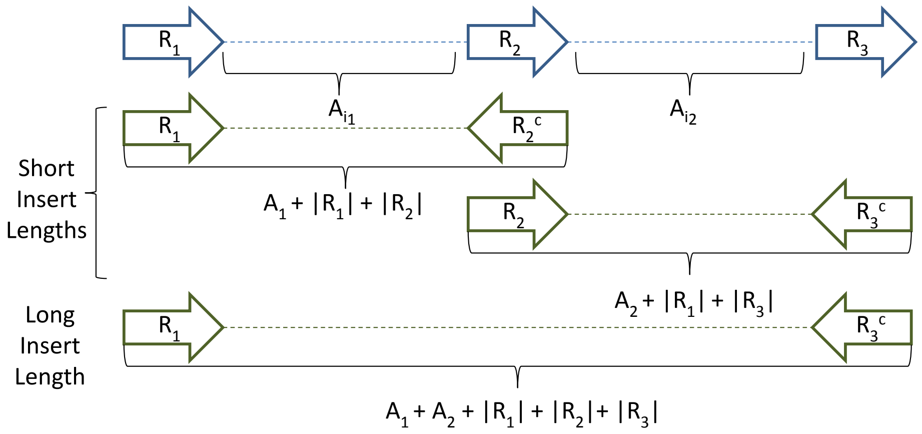

Figure 5.7: **Converting Strobes to Pairs to Preserve Paring Information.** Two paired reads with “short” insert sizes and one paired read with a “long” insert size is created.

Since Velvet is a de-Bruijn graph assembler, at high sequencing error rates we do not expect Velvet to produce accurate assemblies. The highly-fragmented assemblies, even at 5% error, indicates that the seed-based methods perform as expected (Table 5.3). For 5% and 10% error, where smaller k-mer sizes are required to handle the higher error rate, the maximum N50 is 91bp.

	$k = 9$			$k = 11$			$k = 13$		
	#	#$> 2k$	N50	#	#$> 2k$	N50	#	#$> 2k$	N50
B0 1%	44159	6	16	19179	331	30	4583	548	23122
B0 5%	92978	0	13	99565	1230	32	29332	3349	61
B0 10%	110484	0	12	189300	1943	33	55439	10239	91
B1 1%	54024	3	15	26480	356	28	5987	680	19572
B1 5%	102311	0	12	135179	1175	30	39702	4166	62
B1 10%	116621	0	11	252965	1655	31	73150	12689	85
B2 1%	50139	0	15	34208	283	26	8024	795	12034
B2 5%	92813	1	13	156510	987	29	50961	4281	61
B2 10%	108596	0	12	272288	1528	32	94435	12740	84
B3 1%	52013	1	15	29306	304	26	6904	703	23021
B3 5%	97958	1	13	144412	1056	30	43350	4195	61
B3 10%	113599	0	11	263326	1567	31	81010	12693	87

Table 5.3: **Velvet Assembly Results.** We ran Velvet for k-mer size of ($k = 7, 9, 11, 13, 15$); $k = 9, 11, 13$ is shown in this table. We use three statistics to describe the quality of the assembly: the number of contigs (which are nodes in the final graph), the number of contigs that have length greater than $2k$, and the contig N50.

Since Bambus uses an overlap-graph approach to find an assembly, it performs much better than Velvet in assembling the BACs up to 5% error (Table 5.4). However, at 10% error the hash-based aligner produces bad alignments, resulting in a highly-fragmented assembly. To ensure a fair comparison to the assembler portion of Bambus, we removed the aligner from the Bambus pipeline and instead inserted the top 90% optimal alignments used by the ILP Assembler ("Modified Bambus," Table 5.4), corresponding to the set of alignments used to construct the fragment string graph. While Bambus produces reasonable assemblies at 10% error with the optimal alignments, there are still many more contigs and scaffolds than the assemblies produced by the ILP assembler.

The comparison reveals that, even at lower error rates, the ILP assembler is more successful at assembling the BACs than Minimus/Bambus. After scaffolding, Bambus selects a subset of the contigs in the final layout; even this subset of contigs are more fragmented than the ILP assembler (Figure 5.8).

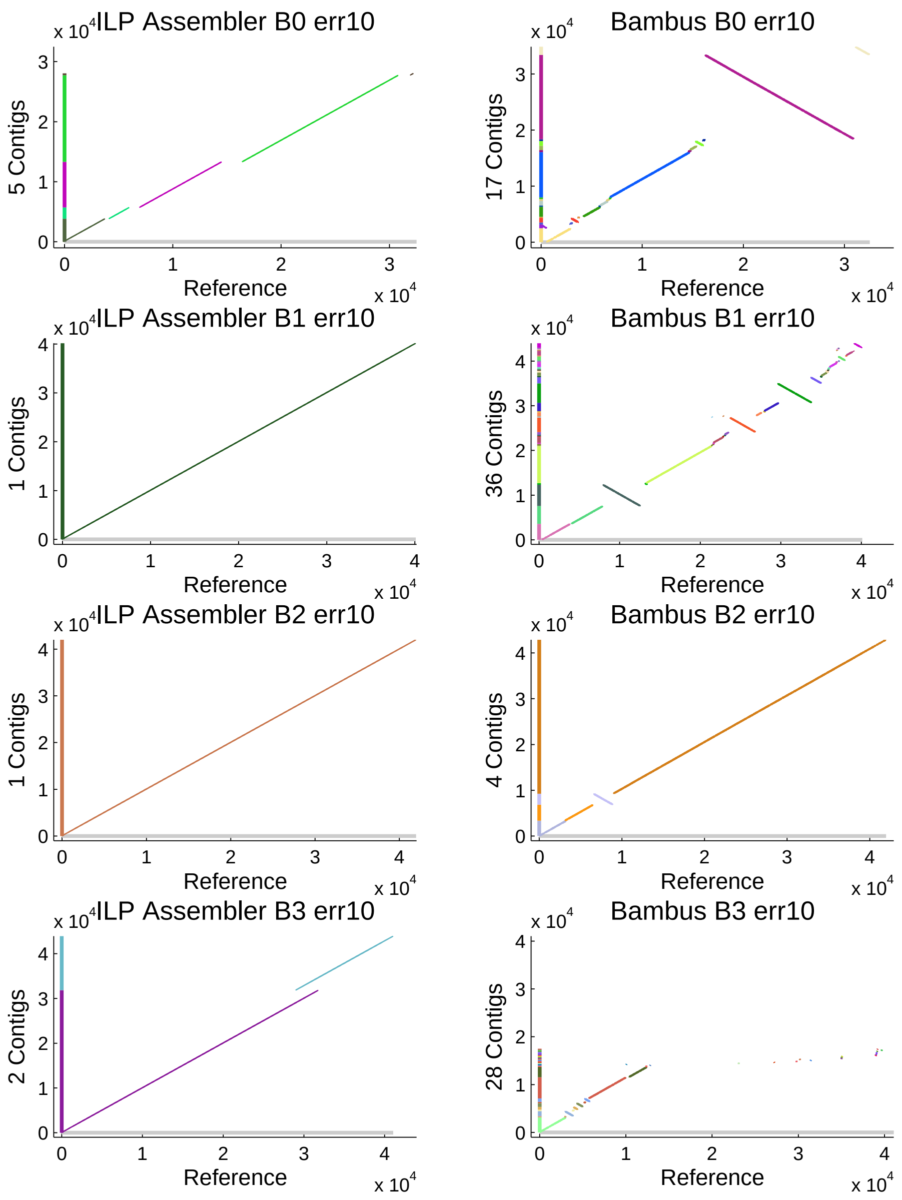

Figure 5.8: **Comparison of Assemblies of BACs with 10% Error.** Each "dot plot" aligns the contigs produced by the assembly to the reference, each colored a different color. Assemblies from the ILP assembler (Left) are more complete (less fragmented) than the assemblies from the Minimus/Bambus Assembler with Smith-Waterman alignments (Right).

	Bambus					Bambus w/ SW Alignments				
	Contigs		Scaffolds		%	Contigs		Scaffolds		%
Dataset	#	N50	#	N50	Assembled	#	N50	#	N50	Assembled
B0 1%	63	8062	1	35187	98%	316	9740	1	31283	93%
B0 5%	281	593	2	49209	94%	324	7977	1	31839	92%
B0 10%	3864	209	81	496	11%	100	7943	1	35337	96%
B1 1%	87	3282	3	45485	96%	96	3931	3	40930	94%
B1 5%	380	504	5	68577	94%	78	3744	1	41969	96%
B1 10%	4757	208	110	483	12%	199	1361	4	43531	92%
B2 1%	17	33027	1	42086	99%	13	33027	1	42126	99%
B2 5%	320	688	2	63515	95%	34	13869	2	42376	99%
B2 10%	4904	209	111	504	13%	20	33016	1	42344	99%
B3 1%	89	3933	3	43056	96%	58	4158	2	43289	98%
B3 5%	289	567	3	61564	95%	62	4053	2	43067	97%
B3 10%	4890	209	100	501	11%	198	1498	9	45244	27%

Table 5.4: **Bambus Results on Tangled Scaffolds.** Original scaffolds produced by Minimus/Bambus are "tangled", and may undergo an optional untangling procedure. Here we report the Bambus results on the tangled scaffolds, because untangling further fractures the assembly resulting in worse N50 values.

5.4 Discussion

In this chapter, we have outlined an ILP-based *de novo* assembler. This assembler optimizes a strobe layout score rather than building a graph and finding a path through this graph. This approach may be beneficial over other, k-mer approaches as read lengths increase and sequencing error rates climb.

We have shown that, in terms of the number of contigs and scaffolds, the ILP assembler outperforms current assemblers on relatively repeat-rich sequences using strobe sequencing data. In particular, it is able to assemble genomes better than Velvet, which is a common assembly method using the de-Bruijn graph, in terms of the contig N50 and the number of contigs produced. Additionally, the ILP assembler produces a smaller number of contigs than Bambus, an overlap-graph assembler that is better designed to handle assembling longer reads with higher error. Further, the assemblies produced by the ILP assembler incorporates a larger portion of the subreads than Bambus, showing that the ILP assembly is more complete.

However, there are many limitations to this method; particularly that it is currently only suitable for small, focused assemblies. That is, much of the assembly must be “easy” to assemble via other methods. In this sense, the ILP assembler can be viewed as an “assembly finisher” to tackle the repeat-rich regions in the assembly.

Additionally, this work conveys the importance of the pairwise alignments used as input to these methods; Bambus, for example, performs quite poorly using their internal aligner on higher error rates, while the assembly construction is agnostic to the aligner used. In Table 5.4, we show that inputting the BLASR alignments to the Bambus framework significantly improves the final assembly. This work has focused on the assembler portion of the process, but careful attention must be paid to the alignments as well.

Chapter 6

Conclusion

It is important to identify and characterize structural variants in human and cancer genomes to gain insight into genetic diversity and disease. Advances in genomic technologies have enabled the identification of a number of structural variants, but these technologies typically require new algorithms to leverage the full benefits of each technology. In this dissertation, we have described a number of technologies and the methods we developed for inferring SVs.

6.1 Summary of Contributions

We have worked with two different types of genomic technologies, aCGH and DNA sequencing, with two different end goals: (1) we inferred recurrent SVs from a group of individuals with the same cancer type, and (2) we inferred SVs from a single individual. NBC generalizes the concept of identifying recurrent intervals to identify recurrent breakpoints, which allows for interchromosomal SV detection. Similarly, the algorithms aimed at DNA sequencing generalize the concept of paired-reads to multiply-linked reads, which allows SV prediction from platforms other than next-generation paired-read sequencers. We briefly summarize the major contributions of this dissertation.

6.1.1 Detecting SVs from aCGH Data

In Chapter 2 we describe NBC [125] to identify recurrent SVs from aCGH data from multiple individuals with the same cancer type. This type of analysis is especially useful in the context of cancer, where we wish to find SVs that may affect the progression of a particular cancer. Rather than identifying recurrent intervals that imply duplications and deletions, NBC finds recurrent *breakpoints* in a probabilistic framework. NBC is the first method for systematically inferring recurrent gene truncations and fusion genes from array copy number data. We applied NBC to data from prostate and glioblastoma tumors, and infer a number of novel fusion genes and gene truncations. These included the known TMPRSS2-ERG fusion gene in prostate cancer, and a number of rearrangements involving the phosphatase PTPN12 in glioblastoma. Since our publication, PTPN12 has been identified as a known oncogene in breast cancer [137].

6.1.2 Detecting SVs from DNA Sequencing Data

With strobe sequencing data from a single individual, we developed the first resequencing methods to identify SVs from multiply-linked reads. The parsimony method [124] in Chapter 3 generalizes a previous problem formulation from [52] and applies it to strobe sequencing data. The probabilistic method in Chapter 4 incorporates the quality of strobe alignments to the reference genome and the expected number of strobes that support an SV, which greatly improves over the parsimony assumption in simulation. Since strobe sequencing has a significantly higher sequencing error rate, both algorithms are designed to handle the large number of ambiguous alignments to the reference genome. In simulation, we demonstrate that low-coverage strobe sequencing outperforms 30X coverage of paired-end sequencing in terms of detectability of SVs and enrichment of correct alignments. Additionally, with the probabilistic method we show that the alignment quality is useful to consider in paired-end sequencing, even though the sequencing error rate is much lower than third generation technologies. Finally, we apply our methods to real strobe sequencing data, including two inversions that are difficult to identify with paired-end sequencing.

Regardless of the DNA sequencing technology, there are still obstacles for SV detection using a resequencing approach because it requires a reference genome, which is problematic for a number of reasons: (1) single nucleotide variants appear as differences between the reference and test genomes, (2) the reference genome is highly-repetitive, so reads will still have ambiguous alignments, and (3) the current version of the human genome draft (hg19) is still incomplete: there are 444 contigs that collectively represent the 23 human chromosomes. An alternate approach, which is becoming more attractive as the read length increases, is to perform a *de novo* assembly of rearranged regions to prevent biases from the reference genome in the SV predictions. In Chapter 5 we present the first steps toward this unbiased type of analysis, where we explicitly use the pairing information as we construct a *de novo* assembly on small regions of the genome. As with methods in the other DNA sequencing chapters, this method generalizes to multiply-linked reads; however, the combinatorial formulation does not yet scale to many cancer genomes and thus our approach is an assembly “finisher” that can be used after the easy-to-assemble portions are complete.

6.2 Future Computational Work and Applications

We designed the algorithms described in this dissertation with specific technologies in mind; NBC was designed for aCGH data and the other algorithms were designed for Pacific Biosciences’ strobe sequencing data. However, the methods themselves are applicable to a variety of technologies. We summarize the overarching themes in this dissertation and describe how our techniques may be used in other settings.

6.2.1 Identifying Recurrent SVs

Identifying recurrent SVs from a group of individuals is an important step for determining SVs that are associated with a particular disease. While we focus on cancer applications, there are other types of case/control applications. Analysis of DNA sequencing data from other genetic diseases such as autism [101, 106] unveil a number of recurrent SVs that may be associated with the disease.

NBC is applied to aCGH data from tumor genomes in our published work, but there are a number of other technologies that can be used to identify recurrent SVs. As discussed in Chapter 2, there are a number of limitations of aCGH data, including (1) the detectability of copy number variants (CNVs) rather than copy-neutral variants, (2) limitations in the placement of probes along the genome, and (3) noise in the measurements themselves. In contrast to aCGH, DNA sequencing provides a much higher resolution of structural variation detection. Unlike array-based methods, shotgun sequencing of DNA provides a sampling of all portions of a test genome, rather than measuring particular probe locations. DNA sequencing technologies are becoming cheap enough that the concept of identifying recurrent SVs among genomes from many individuals is now feasible, and we have begun acquiring a number of DNA sequencing datasets from tumors from The Cancer Genome Atlas with the goal of analyzing them to identify recurrent SVs.

Identifying Recurrent CNVs from Read Depth One signal from DNA sequencing is the number of reads that cover a particular base when aligned to the reference genome, called the *read depth*. Any DNA sequencing protocol, including paired-read and single-read protocols, can be used to estimate read depth. Read depth provides a "copy number profile" similar to the aCGH profiles, but at a much higher resolution. Others have identified recurrent intervals using read depth from DNA sequencing data [28, 157], and this type of data is immediately applicable to NBC. At the time of publication, relatively few tumor genomes had been sequenced, and the small number was not enough to provide statistical significance for the set of inferred CNVs. The Cancer Genome Atlas is currently working to release DNA sequencing data for hundreds of tumors from over twenty different cancer types, so it is now feasible to determine statistically significant breakpoints from read depth data using NBC.

Identifying a Broader Set of Recurrent SVs Using read depth for CNV detection is still limited by the type of SVs detectable (namely duplications and deletions). Other signals from DNA sequencing data, such discordant pairs and split reads, can be used to identify recurrent copy-neutral SVs such as balanced inversions and translocations. An obvious first step is to run the SV detection algorithms on each sample and compare the resulting predicted SVs, but one can imagine more sophisticated methods where all datasets are considered simultaneously when predicting recurrent SVs. The first such formulation was presented in 2011 by [54], but was only demonstrated on mother-father-child trios. More recently, Genome STRiP [49] has identified deletions across hundreds of genomes from the 1000 Genomes Project that were sequenced to low (4x) coverage. For this framework to be applicable to hundreds of high-coverage genomes, the algorithms must be scalable enough to handle potentially terabytes of data.

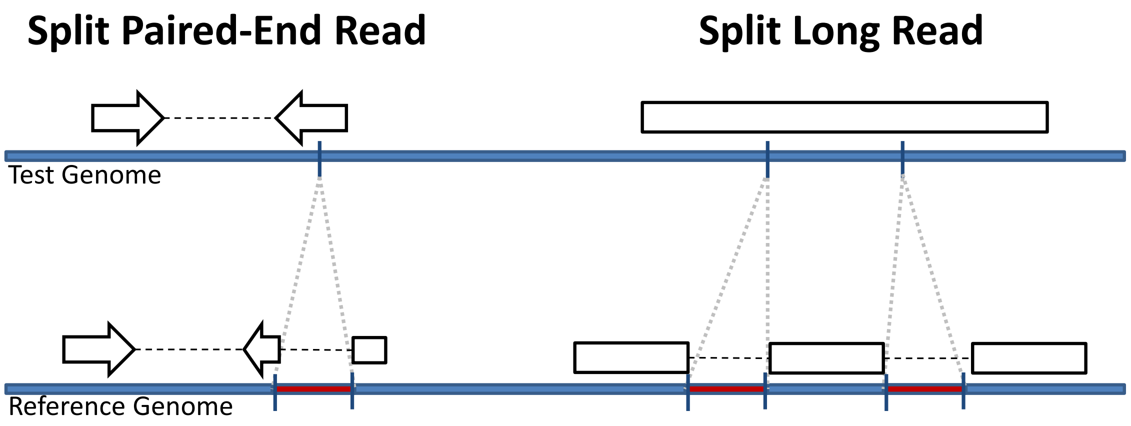

Figure 6.1: **"Virtual Strobes" from Next- and Third-Generation Sequencing Technologies.** (Left) Identifying candidate split reads in paired-read sequencing data produces a virtual strobe with three linked reads. (Right) As third-generation technologies produce longer reads, they might contain multiple SVs. Identifying multiple candidate split reads from long reads produces virtual strobes with any number of linked reads.

6.2.2 Utilizing Multiply-Linked Pairing Information

Incorporating multiply-linked read information (when it is available) improves the performance of SV detection (in Chapter 3) and the ability to assemble highly-repetitive regions (in Chapter 5). We use strobe sequencing as an example of multiply-linked reads, there are a number of other scenarios in which multiple segments of DNA from a single fragment are associated.

Multiply-Linked Reads from Next-Generation Paired-Read Sequencing The read length for paired end sequencing has been steadily increasing; it is currently at 100-150bp for Illumina Hiseq, see Table 1.1. With the increase in read length, there are opportunities to identify reads that span the coordinates of SVs from next-generation sequencing data. For reads that span the precise breakpoints of the SVs, portions of the reads align to disparate locations in the reference genome, resulting in *split reads*. Current methods for split read detection from paired-read sequencing [1, 60, 122, 155] "anchor" a read that is not split in order to identify the paired read that is split. However, when considering paired-end data where one read contains a split, there are three disparate locations that together comprise the true alignment for the pair (Figure 6.1 Left). We can consider this a *virtual strobe* with three subreads, where the advance lengths between the subreads are different.[1] The resequencing methods we developed are robust to a large number of ambiguous alignments. Thus, rather than trying to identify the single correct split read alignment, we can input a set of candidate partial alignments for each read. From these partial alignments we construct a valid set of virtual strobe alignments and run our methods to determine the set of SVs best described by the data.

[1]One of the advances will be the distance between the paired reads, and the other advance will be a single base pair.

Multiply-Linked Reads from Third-Generation Long Read Sequencing While the resequencing methods were designed with strobes in mind, the long reads produced by Pacific Biosciences and nanopore technologies such as Oxford Nanopore also present a relevant application of the "virtual strobe" concept. Pacific Biosciences have published reads as along as 14Kb long, albeit with a 15% error rate. Oxford Nanopore has reportedly sequenced a 48Kb long genome at a 4% error rate (see Table 1.1). As read length increases, a single read may harbor multiple nearby or nested SVs. Again, partial alignments from long reads can constitute a "virtual strobe" of multiply-linked regions, requiring our methods for SV prediction (Figure 6.1 Right). Further, since our methods were designed for Pacific Biosciences data, they are robust to high sequencing error rates of third-generation sequencing technologies.

6.2.3 Utilizing Alignment Quality in SV Detection from DNA Sequencing Data

For resequencing approaches, careful attention must be paid to the read alignments to the genome. Other work has shown that using alignment quality [52, 119] and considering ambiguous alignments to the reference genome [53, 119, 134] improves the detection of SVs from paired-end sequencing data. Our work in Chapter 4 demonstrates that considering multiple alignments along with alignment quality is particularly necessary when considering technologies with higher sequencing error rates. We used PacBio's strobe sequencing to demonstrate this point, but other third-generation sequencing technologies such as Oxford Nanopore report higher error rates than next-generation technologies (see Table 1.1).

6.2.4 Developing Algorithms for Data from Multiple Technologies

Many of the large data centers take multiple measurements of the same sample using different genomic technologies. For example, The Cancer Genome Atlas provides both aCGH and DNA sequencing data for each sample. Data of the same sample from multiple technologies may be combined to predict a more comprehensive set of SVs, or one technology may be used to verify predictions produced by another technology. Additionally, new technologies are often first used to supplement data from existing, more established technologies. This, there is a need for algorithms that are applicable to data from a number of different genomic technologies (such as aCGH and DNA Sequencing) and generalizable for a number of protocols (such as next- and third-Generation sequencing technologies).

The methods described in Chapters 3-5 are immediately applicable to paired-end sequencing because the multiply-linked reads from strobe sequencing is a generalization of paired-end sequencing. Specifically, the probabilistic model for SV detection via resequencing explicitly models the sequence coverage and the sequencing error rate of the DNA technology. However, a number of challenges remain for algorithms that simultaneously analyze data from different technologies. Prior information about different technologies should be considered, including (1) confidence in older, more established technologies, (2) known limitations in the detectability of technologies, and (3) details of the error models produced by different technologies. For example, aCGH data would not be useful to help in the detection of inversions *unless* they are unbalanced inversions where the number of copies of DNA changes. However, if DNA sequencing data implies a duplication or deletion near aCGH probes but the aCGH copy number profile appears to have a normal copy count, then we might conclude that the DNA sequencing data is inferring an incorrect SV.

6.2.5 Additional Uses for the Small-Scale Assembler

The ILP-based assembler presented in Chapter 5 is described as an assembly "finisher," to be used after a partial assembly has been constructed. However, there are other opportunities for improving *de novo* assemblies using the small-scale assembler. Here we describe an application of the small-scale assembler to help resolve ambiguities in overlap graph-based *de novo* assemblers.

In an overlap graph, the edges are contigs and the vertices are junctions where one or more subreads align to multiple contigs, presenting some ambiguity in the assembly. Some methods [23, 24, 110, 111] incorporate pairing information *after* the overlap graph construction; this may resolve some, but not all, of the vertices in the graph. The ILP assembler can help resolve the remaining vertices by considering pairing information and alignment quality simultaneously in the following manner. Suppose we select a vertex in the overlap graph with two incoming and two outgoing edges; the subreads in the vertex belong to some set of strobes. If we assemble the strobe associated with some subread in the vertex, we will get a set of contigs that represent a single optimal layout of the strobes (Figure 6.2). By greedily incorporating strobes whose subreads appear in contigs represented by the adjacent edges, the two paths through the vertex may be distinguishable. Thus, the node is resolved as two paths in the overlap graph. Using an assembly-based approach for node resolution for multiply-linked pairs allows us to use both pairing information and alignment information to resolve the node.

6.3 Future Biological Work and Applications

In this dissertation we present algorithms and analysis on both simulated and real datasets, but the the major contributions are primarily computational. In many ways, the work presented here are "first steps" towards answering much broader questions about genetic diversity in human populations and disease.

Identifying Biological Mechanisms of Structural Variation As we gain a better grasp of the SVs present in a single individual (or recurrent SVs present in multiple individuals), we can begin to investigate the underlying mechanisms and the potential effects of these SVs. For example, Figure 2.11a shows an inferred fusion gene that might arise due to a tandem duplication. We also found support for a Breakage-Fusion-Bridge event, a previously known biological mechanism, detected from DNA sequencing data in an Ovarian tumor [104]. We found data supporting the event by manually looking for inverse tandem duplication signatures in the set of SV predictions. This manual analysis is limiting, however; recently others have begun to model mechanisms like the Breakage-Fusion-Bridge cycle [64].

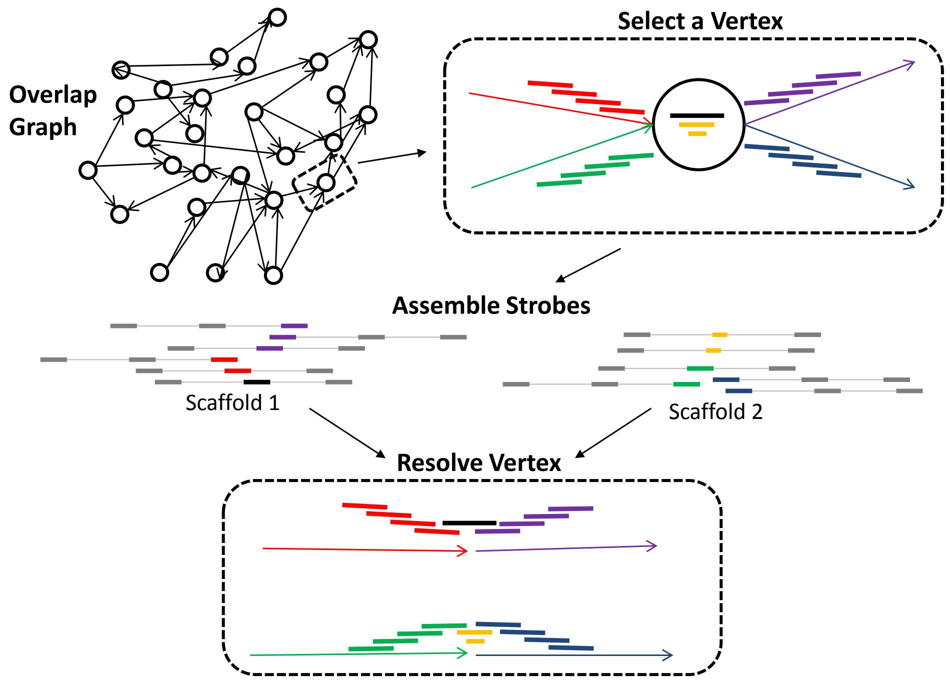

Figure 6.2: **Resolving Vertices in the Overlap Graph.** Assembling the strobes that are associated with subreads from a vertex could potentially resolve the ambiguities represented by the vertex.

Tumor Organization and Progression in Cancer Genomes In cancer, there are a number of driver mutations that promote tumor growth. The temporal order of these driver mutations has been a long-standing question [146], but data limitations have made inference of the order difficult. In particular, a single sample may contain cells from different sub-clonal tumor populations or the cells from the non-tumor genome. Single-cell sequencing strategies [99] have emerged that eliminates the ambiguity of multiple cells sequenced in the same sample, but technological limitations remain [156].

Analysis to infer the genetic composition of tumors has been conducted using a number of different technologies. For example, [103] has recently characterized sub-clonal populations using SNP data from from a single sample with very high (188x) coverage, [42] performed exome sequencing of primary and metastatic renal carcinomas, and [38] combined exome sequencing and copy number data to infer the temporal order in skin and ovarian cancers. Additionally, CGH data [100] and single-cell sequencing data [99] of multiple samples taken from the *same* tumor has shown a large amount of heterogeneity within a single tumor.

Alternate Models of Cancer Progression Finally, we note that the model of cancer progression is not completely understood. The standard assumption is that driver mutations are acquired in an incremental fashion, and take years to accumulate. A recent hypothesis posed about cancer evolution is that, instead of sequentially-acquired mutations, catastrophic events shatter portions of the genome and result in a simultaneous set of driver mutations. This type of event, called chromothripsis, has been observed in a number of cancer types [14, 136]. The improved prediction of “simple” SVs such as duplications, deletions, inversions, and translocations will ultimately allow for the modeling of these more complex events.

Appendices

Appendix A

ILP Assembler Full Formulation

Maximize:

$$\max_{\mathbf{x},\mathbf{y}} Q(\mathcal{L}_{\mathbf{S}})$$

Subject to

Position Constraints:

$$\begin{aligned}
&\forall R \in S, S \in \mathbf{S}: \\
&\quad 0 \le x_R, y_R \le M \\
&\quad |R| - \varepsilon(|R|) - C(1-\eta_S) \le y_R - x_R + 1 \le |R| + \varepsilon(|R|) + C(1-\eta_S) \\
&\quad |R| - \varepsilon(|R|) - C(\eta_S) \le -(y_R - x_R) + 1 \le |R| + \varepsilon(|R|) + C(\eta_S)
\end{aligned}$$

Advance Constraints:

$$\begin{aligned}
&\forall R_i, R_{i+1} \in S, S \in \mathbf{S}: \\
&\quad l - C(1-\eta_S) \le x_{R_{i+1}} - y_{R_i} \le u + C(1-\eta_S) \\
&\quad l - C\eta_S \le -\left(x_{R_{i+1}} - y_{R_i}\right) \le u + C\eta_S
\end{aligned}$$

Overlapping Constraints:

$$\begin{aligned}
&\forall R \in S, R' \in S',\ S, S' \in \mathbf{S}: \\
&\quad (y_R - x_{R'}) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C(1-\eta_S) - C(1-\eta_{S'}) \\
&\quad (y_R - y_{R'}) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C(1-\eta_S) - C\eta_{S'} \\
&\quad (x_R - x_{R'}) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C\eta_S - C(1-\eta_{S'}) \\
&\quad (x_R - y_{R'}) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C\eta_S - C\eta_{S'} \\
&\quad (y_R - x_{R'}) + 1 - \delta \le C\beta_{R,R'} + C\alpha_{R,R'} + C(1-\eta_S) + C(1-\eta_{S'}) - 1 \\
&\quad (y_R - y_{R'}) + 1 - \delta \le C\beta_{R,R'} + C\alpha_{R,R'} + C(1-\eta_S) + C\eta_{S'} - 1 \\
&\quad (x_R - x_{R'}) + 1 - \delta \le C\beta_{R,R'} + C\alpha_{R,R'} + C\eta_S + C(1-\eta_{S'}) - 1 \\
&\quad (x_R - y_{R'}) + 1 - \delta \le C\beta_{R,R'} + C\alpha_{R,R'} + C\eta_S + C\eta_{S'} - 1 \\
&\quad (y_{R'} - x_R) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C(1-\eta_{S'}) - C(1-\eta_S) \\
&\quad (y_{R'} - y_R) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C(1-\eta_{S'}) - C\eta_S \\
&\quad (x_{R'} - x_R) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C\eta_{S'} - C(1-\eta_S) \\
&\quad (x_{R'} - y_R) + 1 - \delta \ge -C(1-\beta_{R,R'}) - C\eta_{S'} - C\eta_S \\
&\quad (y_{R'} - x_R) + 1 - \delta \le C\beta_{R,R'} + C(1-\alpha_{R,R'}) + C(1-\eta_{S'}) + C(1-\eta_S) - 1 \\
&\quad (y_{R'} - y_R) + 1 - \delta \le C\beta_{R,R'} + C(1-\alpha_{R,R'}) + C(1-\eta_{S'}) + C\eta_S - 1 \\
&\quad (x_{R'} - x_R) + 1 - \delta \le C\beta_{R,R'} + C(1-\alpha_{R,R'}) + C\eta_{S'} + C(1-\eta_S) - 1 \\
&\quad (x_{R'} - y_R) + 1 - \delta \le C\beta_{R,R'} + C(1-\alpha_{R,R'}) + C\eta_{S'} + C\eta_S - 1
\end{aligned}$$

Configuration Constraints:

$$\begin{aligned}
&\forall R \in S, R' \in S',\ S, S' \in \mathbf{S}: \\
&\quad x_{R'} - x_R - \mathcal{H}_1(R,R')\eta_S + \mathcal{H}_1(R,R')(1-\eta_S) \ge -C(1-\beta_{R,R'}) - \omega - \varepsilon(\left|\mathcal{H}_1(R,R')\right|) \\
&\quad x_{R'} - x_R - \mathcal{H}_1(R,R')\eta_S + \mathcal{H}_1(R,R')(1-\eta_S) \le C(1-\beta_{R,R'}) + \omega + \varepsilon(\left|\mathcal{H}_1(R,R')\right|) \\
&\quad y_{R'} - y_R - \mathcal{H}_2(R,R')\eta_S + \mathcal{H}_2(R,R')(1-\eta_S) \ge -C(1-\beta_{R,R'}) - \omega - \varepsilon(\left|\mathcal{H}_2(R,R')\right|) \\
&\quad y_{R'} - y_R - \mathcal{H}_2(R,R')\eta_S + \mathcal{H}_2(R,R')(1-\eta_S) \le C(1-\beta_{R,R'}) + \omega + \varepsilon(\left|\mathcal{H}_2(R,R')\right|)
\end{aligned}$$

Bibliography

[1] A. Abyzov and M. Gerstein. AGE: defining breakpoints of genomic structural variants at single-nucleotide resolution, through optimal alignments with gap excision. *Bioinformatics*, 27(5):595–603, Mar 2011.

[2] D. G. Albertson, C. Collins, F. McCormick, and J. W. Gray. Chromosome aberrations in solid tumors. *Nat Genet*, 34(4):369–376, Aug 2003.

[3] C. Alkan, S. Sajjadian, and E. E. Eichler. Limitations of next-generation genome sequence assembly. *Nat Methods*, 8(1):61–65, Jan 2011.

[4] P. N. Ariyaratne and W.-K. Sung. Pe-assembler: de novo assembler using short paired-end reads. *Bioinformatics*, 27(2):167–174, Jan 2011.

[5] A. Baras, Y. Yu, M. Filtz, B. Kim, and C. A. Moskaluk. Combined genomic and gene expression microarray profiling identifies ecop as an upregulated gene in squamous cell carcinomas independent of dna amplification. *Oncogene*, 28(32):2919–2924, Aug 2009.

[6] M. T. Barrett, A. Scheffer, A. Ben-Dor, N. Sampas, D. Lipson, R. Kincaid, P. Tsang, B. Curry, K. Baird, P. S. Meltzer, Z. Yakhini, L. Bruhn, and S. Laderman. Comparative genomic hybridization using oligonucleotide microarrays and total genomic dna. *Proc Natl Acad Sci U S A*, 101(51):17765–17770, Dec 2004.

[7] D. Barry and J. A. Hartigan. A bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309–319, 1993.

[8] A. Bashir, S. Volik, C. Collins, V. Bafna, and B. J. Raphael. Evaluation of paired-end sequencing strategies for detection of genome rearrangements in cancer. *PLoS Comput Biol*, 4(4):e1000051, Apr 2008.

[9] D. Bell, A. Berchuck, M. Birrer, J. Chien, D. Cramer, F. Dao, R. Dhir, P. DiSaia, H. Gabra, P. Glenn, A. Godwin, J. Gross, L. Hartmann, M. Huang, D. Huntsman, M. Iacocca, M. Imielinski, S. Kalloger, B. Karlan, D. Levine, G. Mills, C. Morrison, D. Mutch, N. Olvera, S. Orsulic, K. Park, N. Petrelli, B. Rabeno, J. Rader, B. Sikic, K. Smith-McCune, A. Sood, D. Bowtell, R. Penny, J. Testa, K. Chang,

H. Dinh, J. Drummond, G. Fowler, P. Gunaratne, A. Hawes, C. Kovar, L. Lewis, M. Morgan, I. Newsham, J. Santibanez, J. Reid, L. Trevino, Y. Wu, M. Wang, D. Muzny, D. Wheeler, R. Gibbs, G. Getz, M. Lawrence, K. Cibulskis, A. Sivachenko, C. Sougnez, D. Voet, J. Wilkinson, T. Bloom, K. Ardlie, T. Fennell, J. Baldwin, S. Gabriel, E. Lander, L. L. Ding, R. Fulton, D. Koboldt, M. McLellan, T. Wylie, J. Walker, M. O'Laughlin, D. Dooling, L. Fulton, R. Abbott, N. Dees, Q. Zhang, C. Kandoth, M. Wendl, W. Schierding, D. Shen, C. Harris, H. Schmidt, J. Kalicki, K. Delehaunty, C. Fronick, R. Demeter, L. Cook, J. Wallis, L. Lin, V. Magrini, J. Hodges, J. Eldred, S. Smith, C. Pohl, F. Vandin, B. Raphael, G. Weinstock, E. Mardis, R. Wilson, M. Meyerson, W. Winckler, G. Getz, R. Verhaak, S. Carter, C. Mermel, G. Saksena, H. Nguyen, R. Onofrio, M. Lawrence, D. Hubbard, S. Gupta, A. Crenshaw, A. Ramos, K. Ardlie, L. Chin, A. Protopopov, J. Zhang, T. Kim, I. Perna, Y. Xiao, H. Zhang, G. Ren, N. Sathiamoorthy, R. Park, E. Lee, P. Park, R. Kucherlapati, M. Absher, L. Waite, G. Sherlock, J. Brooks, J. Li, J. Xu, R. Myers, P. W. Laird, L. Cope, J. Herman, H. Shen, D. Weisenberger, H. Noushmehr, F. Pan, T. Triche, B. Berman, D. Van Den Berg, J. Buckley, S. Baylin, P. Spellman, E. Purdom, P. Neuvial, H. Bengtsson, L. Jakkula, S. Durinck, J. Han, S. Dorton, H. Marr, Y. Choi, V. Wang, N. Wang, J. Ngai, J. Conboy, B. Parvin, H. Feiler, T. Speed, J. Gray, A. Levine, N. Socci, Y. Liang, B. Taylor, N. Schultz, L. Borsu, A. Lash, C. Brennan, A. Viale, C. Sander, M. Ladanyi, K. Hoadley, S. Meng, Y. Du, Y. Shi, L. Li, Y. Turman, D. Zang, E. Helms, S. Balu, X. Zhou, J. Wu, M. Topal, D. Hayes, C. Perou, G. Getz, D. Voet, G. Saksena, J. Zhang, H. Zhang, C. Wu, S. Shukla, K. Cibulskis, M. Lawrence, A. Sivachenko, R. Jing, R. Park, Y. Liu, P. Park, M. Noble, L. Chin, H. Carter, D. Kim, R. Karchin, P. Spellman, E. Purdom, P. Neuvial, H. Bengtsson, S. Durinck, J. Han, J. Korkola, L. Heiser, R. Cho, Z. Hu, B. Parvin, T. Speed, J. Gray, N. Schultz, E. Cerami, B. Taylor, A. Olshen, B. Reva, Y. Antipin, R. Shen, P. Mankoo, R. Sheridan, G. Ciriello, W. Chang, J. Bernanke, L. Borsu, D. Levine, M. Ladanyi, C. Sander, D. Haussler, C. Benz, J. Stuart, S. Benz, J. Sanborn, C. Vaske, J. Zhu, C. Szeto, G. Scott, C. Yau, K. Hoadley, Y. Du, S. Balu, D. Hayes, C. Perou, M. Wilkerson, N. Zhang, R. Akbani, K. Baggerly, W. Yung, G. Mills, J. Weinstein, R. Penny, T. Shelton, D. Grimm, M. Hatfield, S. Morris, P. Yena, P. Rhodes, M. Sherman, J. Paulauskis, S. Millis, A. Kahn, J. Greene, R. Sfeir, M. Jensen, J. Chen, J. Whitmore, S. Alonso, J. Jordan, A. Chu, J. Zhang, A. Barker, C. Compton, G. Eley, M. Ferguson, P. Fielding, D. Gerhard, R. Myles, C. Schaefer, K. Mills Shaw, J. Vaught, J. Vockley, P. Good, M. Guyer, B. Ozenberger, J. Peterson, and E. Thomson. Integrated genomic analyses of ovarian carcinoma. *Nature*, 474(7353):609–615, Jun 2011.

[10] A. Ben-Dor, D. Lipson, A. Tsalenko, M. Reimers, L. O. Baumbusch, M. T. Barrett, J. N. Weinstein, A.-L. Børresen-Dale, and Z. Yakhini. Framework for identifying common aberrations in dna copy number data. In *RECOMB*, pages 122–136, 2007.

[11] Y. Benjamini and Y. Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)*,

57(1):289–300, 1995.

[12] D. R. Bentley. Whole-genome re-sequencing. *Curr Opin Genet Dev*, 16(6):545–552, Dec 2006.

[13] D. R. Bentley, S. Balasubramanian, H. P. Swerdlow, G. P. Smith, J. Milton, C. G. Brown, K. P. Hall, D. J. Evers, C. L. Barnes, H. R. Bignell, J. M. Boutell, J. Bryant, R. J. Carter, R. K. Cheetham, A. J. Cox, D. J. Ellis, M. R. Flatbush, N. A. Gormley, S. J. Humphray, L. J. Irving, M. S. Karbelashvili, S. M. Kirk, H. Li, X. Liu, K. S. Maisinger, L. J. Murray, B. Obradovic, T. Ost, M. L. Parkinson, M. R. Pratt, I. M. J. Rasolonjatovo, M. T. Reed, R. Rigatti, C. Rodighiero, M. T. Ross, A. Sabot, S. V. Sankar, A. Scally, G. P. Schroth, M. E. Smith, V. P. Smith, A. Spiridou, P. E. Torrance, S. S. Tzonev, E. H. Vermaas, K. Walter, X. Wu, L. Zhang, M. D. Alam, C. Anastasi, I. C. Aniebo, D. M. D. Bailey, I. R. Bancarz, S. Banerjee, S. G. Barbour, P. A. Baybayan, V. A. Benoit, K. F. Benson, C. Bevis, P. J. Black, A. Boodhun, J. S. Brennan, J. A. Bridgham, R. C. Brown, A. A. Brown, D. H. Buermann, A. A. Bundu, J. C. Burrows, N. P. Carter, N. Castillo, M. C. E. Catenazzi, S. Chang, R. N. Cooley, N. R. Crake, O. O. Dada, K. D. Diakoumakos, B. Dominguez-Fernandez, D. J. Earnshaw, U. C. Egbujor, D. W. Elmore, S. S. Etchin, M. R. Ewan, M. Fedurco, L. J. Fraser, K. V. F. Fajardo, W. S. Furey, D. George, K. J. Gietzen, C. P. Goddard, G. S. Golda, P. A. Granieri, D. E. Green, D. L. Gustafson, N. F. Hansen, K. Harnish, C. D. Haudenschild, N. I. Heyer, M. M. Hims, J. T. Ho, A. M. Horgan, K. Hoschler, S. Hurwitz, D. V. Ivanov, M. Q. Johnson, T. James, T. A. H. Jones, G.-D. Kang, T. H. Kerelska, A. D. Kersey, I. Khrebtukova, A. P. Kindwall, Z. Kingsbury, P. I. Kokko-Gonzales, A. Kumar, M. A. Laurent, C. T. Lawley, S. E. Lee, X. Lee, A. K. Liao, J. A. Loch, M. Lok, S. Luo, R. M. Mammen, J. W. Martin, P. G. McCauley, P. McNitt, P. Mehta, K. W. Moon, J. W. Mullens, T. Newington, Z. Ning, B. L. Ng, S. M. Novo, M. J. O'Neill, M. A. Osborne, A. Osnowski, O. Ostadan, L. L. Paraschos, L. Pickering, A. C. Pike, A. C. Pike, D. C. Pinkard, D. P. Pliskin, J. Podhasky, V. J. Quijano, C. Raczy, V. H. Rae, S. R. Rawlings, A. C. Rodriguez, P. M. Roe, J. Rogers, M. C. R. Bacigalupo, N. Romanov, A. Romieu, R. K. Roth, N. J. Rourke, S. T. Ruediger, E. Rusman, R. M. Sanches-Kuiper, M. R. Schenker, J. M. Seoane, R. J. Shaw, M. K. Shiver, S. W. Short, N. L. Sizto, J. P. Sluis, M. A. Smith, J. E. S. Sohna, E. J. Spence, K. Stevens, N. Sutton, L. Szajkowski, C. L. Tregidgo, G. Turcatti, S. Vandevondele, Y. Verhovsky, S. M. Virk, S. Wakelin, G. C. Walcott, J. Wang, G. J. Worsley, J. Yan, L. Yau, M. Zuerlein, J. Rogers, J. C. Mullikin, M. E. Hurles, N. J. McCooke, J. S. West, F. L. Oaks, P. L. Lundberg, D. Klenerman, R. Durbin, and A. J. Smith. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456(7218):53–59, Nov 2008.

[14] M. F. Berger, M. S. Lawrence, F. Demichelis, Y. Drier, K. Cibulskis, A. Y. Sivachenko, A. Sboner, R. Esgueva, D. Pflueger, C. Sougnez, R. Onofrio, S. L. Carter, K. Park, L. Habegger, L. Ambrogio, T. Fennell, M. Parkin, G. Saksena, D. Voet, A. H. Ramos, T. J. Pugh, J. Wilkinson, S. Fisher, W. Winckler, S. Mahan, K. Ardlie, J. Baldwin, J. W. Simons, N. Kitabayashi, T. Y. MacDonald, P. W.

Kantoff, L. Chin, S. B. Gabriel, M. B. Gerstein, T. R. Golub, M. Meyerson, A. Tewari, E. S. Lander, G. Getz, M. A. Rubin, and L. A. Garraway. The genomic complexity of primary human prostate cancer. *Nature*, 470(7333):214–220, Feb 2011.

[15] R. Beroukhim, G. Getz, L. Nghiemphu, J. Barretina, T. Hsueh, D. Linhart, I. Vivanco, J. C. Lee, J. H. Huang, S. Alexander, J. Du, T. Kau, R. K. Thomas, K. Shah, H. Soto, S. Perner, J. Prensner, R. M. Debiasi, F. Demichelis, C. Hatton, M. A. Rubin, L. A. Garraway, S. F. Nelson, L. Liau, P. S. Mischel, T. F. Cloughesy, M. Meyerson, T. A. Golub, E. S. Lander, I. K. Mellinghoff, and W. R. Sellers. Assessing the significance of chromosomal aberrations in cancer: methodology and application to glioma. *Proc Natl Acad Sci U S A*, 104(50):20007–20012, Dec 2007.

[16] D. W. Bryant, W.-K. Wong, and T. C. Mockler. Qsra: a quality-value guided de novo short read assembler. *BMC Bioinformatics*, 10:69, 2009.

[17] J. Butler, I. MacCallum, M. Kleber, I. A. Shlyakhter, M. K. Belmonte, E. S. Lander, C. Nusbaum, and D. B. Jaffe. Allpaths: de novo assembly of whole-genome shotgun microreads. *Genome Res*, 18(5):810–820, May 2008.

[18] P. J. Campbell, P. J. Stephens, E. D. Pleasance, S. O'Meara, H. Li, T. Santarius, L. A. Stebbings, C. Leroy, S. Edkins, C. Hardy, J. W. Teague, A. Menzies, I. Goodhead, D. J. Turner, C. M. Clee, M. A. Quail, A. Cox, C. Brown, R. Durbin, M. E. Hurles, P. A. W. Edwards, G. R. Bignell, M. R. Stratton, and P. A. Futreal. Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing. *Nat Genet*, 40(6):722–729, Jun 2008.

[19] M. M. Carrasquillo, F. Zou, V. S. Pankratz, S. L. Wilcox, L. Ma, L. P. Walker, S. G. Younkin, C. S. Younkin, L. H. Younkin, G. D. Bisceglio, N. Ertekin-Taner, J. E. Crook, D. W. Dickson, R. C. Petersen, N. R. Graff-Radford, and S. G. Younkin. Genetic variation in pcdh11x is associated with susceptibility to late-onset alzheimer's disease. *Nat Genet*, 41(2):192–198, Feb 2009.

[20] M. Chaisson. Alignment of long reads with indel error using BLASR - Branched Local Alignment with Successive Refinement. *In preparation*, 2010.

[21] M. Chaisson. Alchemy. https://github.com/PacificBiosciences/blasr/tree/master/simulator, 2012.

[22] M. Chaisson. Alignment of long reads with indel error using BLASR - Branched Local Alignment with Successive Refinement. *In preparation*, 2012.

[23] M. J. Chaisson, D. Brinza, and P. A. Pevzner. De novo fragment assembly with short mate-paired reads: Does the read length matter? *Genome Res*, 19(2):336–346, Feb 2009.

[24] M. J. Chaisson and P. A. Pevzner. Short read fragment assembly of bacterial genomes. *Genome Res*, 18(2):324–330, Feb 2008.

[25] M.-C. Chartier-Harlin, J. Kachergus, C. Roumier, V. Mouroux, X. Douay, S. Lincoln, C. Levecque, L. Larvor, J. Andrieux, M. Hulihan, N. Waucquier, L. Defebvre, P. Amouyel, M. Farrer, and A. Deste. Alpha-synuclein locus duplication as a cause of familial parkinson's disease. *Lancet*, 364(9440):1167–1169, 2004.

[26] K. Chen, J. W. Wallis, M. D. McLellan, D. E. Larson, J. M. Kalicki, C. S. Pohl, S. D. McGrath, M. C. Wendl, Q. Zhang, D. P. Locke, X. Shi, R. S. Fulton, T. J. Ley, R. K. Wilson, L. Ding, and E. R. Mardis. Breakdancer: an algorithm for high-resolution mapping of genomic structural variation. *Nat Methods*, 6(9):677–681, Sep 2009.

[27] B. Chevreux. *MIRA: An Automated Genome and EST Assembler*. PhD Thesis, The Ruprecht-Karls-University, 2005.

[28] D. Y. Chiang, G. Getz, D. B. Jaffe, M. J. T. O'Kelly, X. Zhao, S. L. Carter, C. Russ, C. Nusbaum, M. Meyerson, and E. S. Lander. High-resolution mapping of copy-number alterations with massively parallel sequencing. *Nat Methods*, 6(1):99–103, Jan 2009.

[29] C.-S. Chin, J. Sorenson, J. B. Harris, W. P. Robins, R. C. Charles, R. R. Jean-Charles, J. Bullard, D. R. Webster, A. Kasarskis, P. Peluso, E. E. Paxinos, Y. Yamaichi, S. B. Calderwood, J. J. Mekalanos, E. E. Schadt, and M. K. Waldor. The origin of the haitian cholera outbreak strain. *New England Journal of Medicine*, 364(1):33–42, 2011.

[30] K. W. Choy, S. R. Setlur, C. Lee, and T. K. Lau. The impact of human copy number variation on a new era of genetic testing. *BJOG*, 117(4):391–398, Mar 2010.

[31] W. G. Christen, R. J. Glynn, E. Y. Chew, C. M. Albert, and J. E. Manson. Folic acid, pyridoxine, and cyanocobalamin combination treatment and age-related macular degeneration in women: the women's antioxidant and folic acid cardiovascular study. *Arch Intern Med*, 169(4):335–341, Feb 2009.

[32] . G. P. Consortium, R. M. Durbin, G. R. Abecasis, D. L. Altshuler, A. Auton, L. D. Brooks, R. M. Durbin, R. A. Gibbs, M. E. Hurles, and G. A. McVean. A map of human genome variation from population-scale sequencing. *Nature*, 467(7319):1061–1073, Oct 2010.

[33] I. H. Consortium. The international hapmap project. *Nature*, 426(6968):789–796, Dec 2003.

[34] I. H. Consortium. A haplotype map of the human genome. *Nature*, 437(7063):1299–1320, Oct 2005.

[35] T. Crainic, A. Frangioni, and B. Gendron. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112(1-3):73–99, 2001.

[36] H. David and H. Nagaraja. *Order Statistics*. John Wiley, Hoboken,NJ, 3rd edition, 2003.

[37] S. J. Diskin, T. Eck, J. Greshock, Y. P. Mosse, T. Naylor, C. J. Stoeckert, B. L. Weber, J. M. Maris, and G. R. Grant. Stac: A method for testing the significance of dna copy number aberrations across multiple array-cgh experiments. *Genome Res*, 16(9):1149–1158, Sep 2006.

[38] S. Durinck, C. Ho, N. J. Wang, W. Liao, L. R. Jakkula, E. A. Collisson, J. Pons, S. W. Chan, E. T. Lam, C. Chu, K. Park, S. W. Hong, J. S. Hur, N. Huh, I. M. Neuhaus, S. S. Yu, R. C. Grekin, T. M. Mauro, J. E. Cleaver, P. Y. Kwok, P. E. LeBoit, G. Getz, K. Cibulskis, J. C. Aster, H. Huang, E. Purdom, J. Li, L. Bolund, S. T. Arron, J. W. Gray, P. T. Spellman, and R. J. Cho. Temporal dissection of tumorigenesis in primary cancers. *Cancer Discov*, 1(2):137–143, Jul 2011.

[39] J. Eid, A. Fehr, J. Gray, K. Luong, J. Lyle, G. Otto, P. Peluso, D. Rank, P. Baybayan, B. Bettman, A. Bibillo, K. Bjornson, B. Chaudhuri, F. Christians, R. Cicero, S. Clark, R. Dalal, A. Dewinter, J. Dixon, M. Foquet, A. Gaertner, P. Hardenbol, C. Heiner, K. Hester, D. Holden, G. Kearns, X. Kong, R. Kuse, Y. Lacroix, S. Lin, P. Lundquist, C. Ma, P. Marks, M. Maxham, D. Murphy, I. Park, T. Pham, M. Phillips, J. Roy, R. Sebra, G. Shen, J. Sorenson, A. Tomaney, K. Travers, M. Trulson, J. Vieceli, J. Wegener, D. Wu, A. Yang, D. Zaccarin, P. Zhao, F. Zhong, J. Korlach, and S. Turner. Real-time dna sequencing from single polymerase molecules. *Science*, 323(5910):133–138, Jan 2009.

[40] G. D. Eley, J. L. Reiter, A. Pandita, S. Park, R. B. Jenkins, N. J. Maihle, and C. D. James. A chromosomal region 7p11.2 transcript map: its development and application to the study of egfr amplicons in glioblastoma. *Neuro Oncol*, 4(2):86–94, Apr 2002.

[41] C. Erdman and J. W. Emerson. A fast bayesian change point analysis for the segmentation of microarray data. *Bioinformatics*, 24(19):2143–2148, Oct 2008.

[42] M. Gerlinger, A. J. Rowan, S. Horswell, J. Larkin, D. Endesfelder, E. Gronroos, P. Martinez, N. Matthews, A. Stewart, P. Tarpey, I. Varela, B. Phillimore, S. Begum, N. Q. McDonald, A. Butler, D. Jones, K. Raine, C. Latimer, C. R. Santos, M. Nohadani, A. C. Eklund, B. Spencer-Dene, G. Clark, L. Pickering, G. Stamp, M. Gore, Z. Szallasi, J. Downward, P. A. Futreal, and C. Swanton. Intratumor heterogeneity and branched evolution revealed by multiregion sequencing. *N. Engl. J. Med.*, 366(10):883–892, Mar 2012.

[43] A. Gilles, E. Meglecz, N. Pech, S. Ferreira, T. Malausa, and J. F. Martin. Accuracy and quality assessment of 454 GS-FLX Titanium pyrosequencing. *BMC Genomics*, 12:245, 2011.

[44] D. Gisselsson. Mechanisms of whole chromosome gains in tumors - many answers to a simple question. *Cytogenet Genome Res*, Dec 2010.

[45] S. Gnerre, I. Maccallum, D. Przybylski, F. J. Ribeiro, J. N. Burton, B. J. Walker, T. Sharpe, G. Hall, T. P. Shea, S. Sykes, A. M. Berlin, D. Aird, M. Costello, R. Daza, L. Williams, R. Nicol, A. Gnirke,

C. Nusbaum, E. S. Lander, and D. B. Jaffe. High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc Natl Acad Sci U S A*, 108(4):1513–1518, Jan 2011.

[46] Y. H. Grad, M. Lipsitch, M. Feldgarden, H. M. Arachchi, G. C. Cerqueira, M. Fitzgerald, P. Godfrey, B. J. Haas, C. I. Murphy, C. Russ, S. Sykes, B. J. Walker, J. R. Wortman, S. Young, Q. Zeng, A. Abouelleil, J. Bochicchio, S. Chauvin, T. Desmet, S. Gujja, C. McCowan, A. Montmayeur, S. Steelman, J. Frimodt-Muller, A. M. Petersen, C. Struve, K. A. Krogfelt, E. Bingen, F. X. Weill, E. S. Lander, C. Nusbaum, B. W. Birren, D. T. Hung, and W. P. Hanage. Genomic epidemiology of the Escherichia coli O104:H4 outbreaks in Europe, 2011. *Proc. Natl. Acad. Sci. U.S.A.*, 109(8):3065–3070, Feb 2012.

[47] S. Guha, Y. Li, and D. Neuberg. Bayesian hidden markov modeling of array cgh data. *Journal of the American Statistical Association*, 103:485–497, June 2008.

[48] D. Gusfield. *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*. Cambridge University Press, 1997.

[49] R. E. Handsaker, J. M. Korn, J. Nemesh, and S. A. McCarroll. Discovery and genotyping of genome structural polymorphism by sequencing on a population scale. *Nat. Genet.*, 43(3):269–276, Mar 2011.

[50] D. Hernandez, P. Franois, L. Farinelli, M. Osters, and J. Schrenzel. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome Res*, 18(5):802–809, May 2008.

[51] D. Hochbaum and A. Segev. Analysis of a flow problem with fixed charges. *Networks*, 19(3):291–312, 1989.

[52] F. Hormozdiari, C. Alkan, E. E. Eichler, and S. C. Sahinalp. Combinatorial algorithms for structural variation detection in high-throughput sequenced genomes. *Genome Res*, 19(7):1270–1278, Jul 2009.

[53] F. Hormozdiari, I. Hajirasouliha, P. Dao, F. Hach, D. Yorukoglu, C. Alkan, E. E. Eichler, and S. C. Sahinalp. Next-generation variationhunter: combinatorial algorithms for transposon insertion discovery. *Bioinformatics*, 26(12):i350–i357, Jun 2010.

[54] F. Hormozdiari, I. Hajirasouliha, A. McPherson, E. E. Eichler, and S. C. Sahinalp. Simultaneous structural variation discovery among multiple paired-end sequenced genomes. *Genome Res.*, 21(12):2203–2212, Dec 2011.

[55] C. R. Huang, A. M. Schneider, Y. Lu, T. Niranjan, P. Shen, M. A. Robinson, J. P. Steranka, D. Valle, C. I. Civin, T. Wang, S. J. Wheelan, H. Ji, J. D. Boeke, and K. H. Burns. Mobile interspersed repeats are major structural variants in the human genome. *Cell*, 141(7):1171–1182, Jun 2010.

[56] P. Hup, N. Stransky, J.-P. Thiery, F. Radvanyi, and E. Barillot. Analysis of array cgh data: from signal ratio to gain and loss of dna regions. *Bioinformatics*, 20(18):3413–3422, Dec 2004.

[57] K. Huse, S. Taudien, M. Groth, P. Rosenstiel, K. Szafranski, M. Hiller, J. Hampe, K. Junker, J. Schubert, S. Schreiber, G. Birkenmeier, M. Krawczak, and M. Platzer. Genetic variants of the copy number polymorphic beta-defensin locus are associated with sporadic prostate cancer. *Tumour Biol*, 29(2):83–92, 2008.

[58] A. J. Iafrate, L. Feuk, M. N. Rivera, M. L. Listewnik, P. K. Donahoe, Y. Qi, S. W. Scherer, and C. Lee. Detection of large-scale variation in the human genome. *Nat Genet*, 36(9):949–951, Sep 2004.

[59] W. R. Jeck, J. A. Reinhardt, D. A. Baltrus, M. T. Hickenbotham, V. Magrini, E. R. Mardis, J. L. Dangl, and C. D. Jones. Extending assembly of short dna sequences to handle error. *Bioinformatics*, 23(21):2942–2944, Nov 2007.

[60] Y. Jiang, Y. Wang, and M. Brudno. Prism: Pair read informed split read mapping for base-pair level detection of insertion, deletion and structural variants. *Bioinformatics*, 2012.

[61] N. C. Jones and P. A. Pevzner. *An Introduction to Bioinformatics Algorithms (Computational Molecular Biology)*. The MIT Press, Aug. 2004.

[62] J. M. Kidd, G. M. Cooper, W. F. Donahue, H. S. Hayden, N. Sampas, T. Graves, N. Hansen, B. Teague, C. Alkan, F. Antonacci, E. Haugen, T. Zerr, N. A. Yamada, P. Tsang, T. L. Newman, E. Tzn, Z. Cheng, H. M. Ebling, N. Tusneem, R. David, W. Gillett, K. A. Phelps, M. Weaver, D. Saranga, A. Brand, W. Tao, E. Gustafson, K. McKernan, L. Chen, M. Malig, J. D. Smith, J. M. Korn, S. A. McCarroll, D. A. Altshuler, D. A. Peiffer, M. Dorschner, J. Stamatoyannopoulos, D. Schwartz, D. A. Nickerson, J. C. Mullikin, R. K. Wilson, L. Bruhn, M. V. Olson, R. Kaul, D. R. Smith, and E. E. Eichler. Mapping and sequencing of structural variation from eight human genomes. *Nature*, 453(7191):56–64, May 2008.

[63] J. M. Kidd, N. Sampas, F. Antonacci, T. Graves, R. Fulton, H. S. Hayden, C. Alkan, M. Malig, M. Ventura, G. Giannuzzi, J. Kallicki, P. Anderson, A. Tsalenko, N. A. Yamada, P. Tsang, R. Kaul, R. K. Wilson, L. Bruhn, and E. E. Eichler. Characterization of missing human genome sequences and copy-number polymorphic insertions. *Nat Methods*, 7(5):365–371, May 2010.

[64] M. Kinsella and V. Bafna. Modeling the breakage-fusion-bridge mechanism: Combinatorics and cancer genomics. In *RECOMB'12*, pages 148–162, 2012.

[65] J. O. Korbel, A. Abyzov, X. J. Mu, N. Carriero, P. Cayting, Z. Zhang, M. Snyder, and M. B. Gerstein. Pemer: a computational framework with simulation-based error models for inferring genomic structural variants from massive paired-end sequencing data. *Genome Biol*, 10(2):R23, 2009.

[66] J. O. Korbel, A. E. Urban, J. P. Affourtit, B. Godwin, F. Grubert, J. F. Simons, P. M. Kim, D. Palejev, N. J. Carriero, L. Du, B. E. Taillon, Z. Chen, A. Tanzer, A. C. E. Saunders, J. Chi, F. Yang, N. P. Carter, M. E. Hurles, S. M. Weissman, T. T. Harkins, M. B. Gerstein, M. Egholm, and M. Snyder. Paired-end mapping reveals extensive structural variation in the human genome. *Science*, 318(5849):420–426, Oct 2007.

[67] J. Korlach, K. P. Bjornson, B. P. Chaudhuri, R. L. Cicero, B. A. Flusberg, J. J. Gray, D. Holden, R. Saxena, J. Wegener, and S. W. Turner. Real-time dna sequencing from single polymerase molecules. *Methods Enzymol*, 472:431–455, 2010.

[68] P. Krawitz, C. Rdelsperger, M. Jger, L. Jostins, S. Bauer, and P. N. Robinson. Microindel detection in short-read sequence data. *Bioinformatics*, 26(6):722–729, Mar 2010.

[69] W. R. Lai, M. D. Johnson, R. Kucherlapati, and P. J. Park. Comparative analysis of algorithms for identifying amplifications and deletions in array cgh data. *Bioinformatics*, 21(19):3763–3770, Oct 2005.

[70] H. Y. Lam, M. J. Clark, R. Chen, R. Chen, G. Natsoulis, M. O'Huallachain, F. E. Dewey, L. Habegger, E. A. Ashley, M. B. Gerstein, A. J. Butte, H. P. Ji, and M. Snyder. Performance comparison of whole-genome sequencing platforms. *Nat. Biotechnol.*, 30(1):78–82, Jan 2012.

[71] J. Laserson, V. Jojic, and D. Koller. Genovo: de novo assembly for metagenomes. *J Comput Biol*, 18(3):429–443, Mar 2011.

[72] S. Lee, E. Cheran, and M. Brudno. A robust framework for detecting structural variations in a genome. *Bioinformatics*, 24(13):i59–i67, Jul 2008.

[73] S. Levy, G. Sutton, P. C. Ng, L. Feuk, A. L. Halpern, B. P. Walenz, N. Axelrod, J. Huang, E. F. Kirkness, G. Denisov, Y. Lin, J. R. MacDonald, A. W. Pang, M. Shago, T. B. Stockwell, A. Tsiamouri, V. Bafna, V. Bansal, S. A. Kravitz, D. A. Busam, K. Y. Beeson, T. C. McIntosh, K. A. Remington, J. F. Abril, J. Gill, J. Borman, Y. H. Rogers, M. E. Frazier, S. W. Scherer, R. L. Strausberg, and J. C. Venter. The diploid genome sequence of an individual human. *PLoS Biol.*, 5(10):e254, Sep 2007.

[74] H. Li and R. Durbin. Fast and accurate short read alignment with burrows-wheeler transform. *Bioinformatics*, 25(14):1754–1760, Jul 2009.

[75] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and . G. P. D. P. Subgroup. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, Aug 2009.

[76] R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, S. Li, H. Yang, J. Wang, and J. Wang. De novo assembly of human genomes with massively parallel short read sequencing. *Genome Res*, 20(2):265–272, Feb 2010.

[77] D. Lipson, Y. Aumann, A. Ben-Dor, N. Linial, and Z. Yakhini. Efficient calculation of interval scores for dna copy number data analysis. *J Comput Biol*, 13(2):215–228, Mar 2006.

[78] J. S. Liu and C. E. Lawrence. Bayesian inference on biopolymer models. *Bioinformatics*, 15(1):38–52, Jan 1999.

[79] L. Liu, Y. Li, S. Li, N. Hu, Y. He, R. Pong, D. Lin, L. Lu, and M. Law. Comparison of next-generation sequencing systems. *J. Biomed. Biotechnol.*, 2012:251364, 2012.

[80] N. J. Loman, R. V. Misra, T. J. Dallman, C. Constantinidou, S. E. Gharbia, J. Wain, and M. J. Pallen. Performance comparison of benchtop high-throughput sequencing platforms. *Nat. Biotechnol.*, 30(5):434–439, May 2012.

[81] R. Lucito and J. Byrnes. Comparative genomic hybridization by representational oligonucleotide microarray analysis. *Methods Mol Biol*, 556:33–46, 2009.

[82] J. R. Lupski, J. G. Reid, C. Gonzaga-Jauregui, D. Rio Deiros, D. C. Chen, L. Nazareth, M. Bainbridge, H. Dinh, C. Jing, D. A. Wheeler, A. L. McGuire, F. Zhang, P. Stankiewicz, J. J. Halperin, C. Yang, C. Gehman, D. Guo, R. K. Irikat, W. Tom, N. J. Fantin, D. M. Muzny, and R. A. Gibbs. Whole-genome sequencing in a patient with charcot-marie-tooth neuropathy. *New England Journal of Medicine*, 362(13):1181–1191, 2010.

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

[84] C. R. Marshall, A. Noor, J. B. Vincent, A. C. Lionel, L. Feuk, J. Skaug, M. Shago, R. Moessner, D. Pinto, Y. Ren, B. Thiruvahindrapduram, A. Fiebig, S. Schreiber, J. Friedman, C. E. J. Ketelaars, Y. J. Vos, C. Ficicioglu, S. Kirkpatrick, R. Nicolson, L. Sloman, A. Summers, C. A. Gibbons, A. Teebi, D. Chitayat, R. Weksberg, A. Thompson, C. Vardy, V. Crosbie, S. Luscombe, R. Baatjes, L. Zwaigenbaum, W. Roberts, B. Fernandez, P. Szatmari, and S. W. Scherer. Structural variation of chromosomes in autism spectrum disorder. *Am J Hum Genet*, 82(2):477–488, Feb 2008.

[85] R. McLendon, A. Friedman, D. Bigner, E. G. Van Meir, D. J. Brat, G. M. Mastrogianakis, J. J. Olson, T. Mikkelsen, N. Lehman, K. Aldape, W. K. Yung, O. Bogler, J. N. Weinstein, S. VandenBerg, M. Berger, M. Prados, D. Muzny, M. Morgan, S. Scherer, A. Sabo, L. Nazareth, L. Lewis, O. Hall, Y. Zhu, Y. Ren, O. Alvi, J. Yao, A. Hawes, S. Jhangiani, G. Fowler, A. San Lucas, C. Kovar, A. Cree, H. Dinh, J. Santibanez, V. Joshi, M. L. Gonzalez-Garay, C. A. Miller, A. Milosavljevic, L. Donehower, D. A. Wheeler, R. A. Gibbs, K. Cibulskis, C. Sougnez, T. Fennell, S. Mahan, J. Wilkinson, L. Ziaugra, R. Onofrio, T. Bloom, R. Nicol, K. Ardlie, J. Baldwin, S. Gabriel, E. S. Lander, L. Ding, R. S. Fulton, M. D. McLellan, J. Wallis, D. E. Larson, X. Shi, R. Abbott, L. Fulton, K. Chen, D. C. Koboldt, M. C. Wendl, R. Meyer, Y. Tang, L. Lin, J. R. Osborne, B. H. Dunford-Shore, T. L. Miner,

K. Delehaunty, C. Markovic, G. Swift, W. Courtney, C. Pohl, S. Abbott, A. Hawkins, S. Leong, C. Haipek, H. Schmidt, M. Wiechert, T. Vickery, S. Scott, D. J. Dooling, A. Chinwalla, G. M. Weinstock, E. R. Mardis, R. K. Wilson, G. Getz, W. Winckler, R. G. Verhaak, M. S. Lawrence, M. O'Kelly, J. Robinson, G. Alexe, R. Beroukhim, S. Carter, D. Chiang, J. Gould, S. Gupta, J. Korn, C. Mermel, J. Mesirov, S. Monti, H. Nguyen, M. Parkin, M. Reich, N. Stransky, B. A. Weir, L. Garraway, T. Golub, M. Meyerson, L. Chin, A. Protopopov, J. Zhang, I. Perna, S. Aronson, N. Sathiamoorthy, G. Ren, J. Yao, W. R. Wiedemeyer, H. Kim, S. W. Kong, Y. Xiao, I. S. Kohane, J. Seidman, P. J. Park, R. Kucherlapati, P. W. Laird, L. Cope, J. G. Herman, D. J. Weisenberger, F. Pan, D. Van den Berg, L. Van Neste, J. M. Yi, K. E. Schuebel, S. B. Baylin, D. M. Absher, J. Z. Li, A. Southwick, S. Brady, A. Aggarwal, T. Chung, G. Sherlock, J. D. Brooks, R. M. Myers, P. T. Spellman, E. Purdom, L. R. Jakkula, A. V. Lapuk, H. Marr, S. Dorton, Y. G. Choi, J. Han, A. Ray, V. Wang, S. Durinck, M. Robinson, N. J. Wang, K. Vranizan, V. Peng, E. Van Name, G. V. Fontenay, J. Ngai, J. G. Conboy, B. Parvin, H. S. Feiler, T. P. Speed, J. W. Gray, C. Brennan, N. D. Socci, A. Olshen, B. S. Taylor, A. Lash, N. Schultz, B. Reva, Y. Antipin, A. Stukalov, B. Gross, E. Cerami, W. Q. Wang, L. X. Qin, V. E. Seshan, L. Villafania, M. Cavatore, L. Borsu, A. Viale, W. Gerald, C. Sander, M. Ladanyi, C. M. Perou, D. N. Hayes, M. D. Topal, K. A. Hoadley, Y. Qi, S. Balu, Y. Shi, J. Wu, R. Penny, M. Bittner, T. Shelton, E. Lenkiewicz, S. Morris, D. Beasley, S. Sanders, A. Kahn, R. Sfeir, J. Chen, D. Nassau, L. Feng, E. Hickey, A. Barker, D. S. Gerhard, J. Vockley, C. Compton, J. Vaught, P. Fielding, M. L. Ferguson, C. Schaefer, J. Zhang, S. Madhavan, K. H. Buetow, F. Collins, P. Good, M. Guyer, B. Ozenberger, J. Peterson, and E. Thomson. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216):1061–1068, Oct 2008.

[86] P. Medvedev and M. Brudno. Maximum likelihood genome assembly. *J Comput Biol*, 16(8):1101–1116, Aug 2009.

[87] A. Mellmann, D. Harmsen, C. A. Cummings, E. B. Zentz, S. R. Leopold, A. Rico, K. Prior, R. Szczepanowski, Y. Ji, W. Zhang, S. F. McLaughlin, J. K. Henkhaus, B. Leopold, M. Bielaszewska, R. Prager, P. M. Brzoska, R. L. Moore, S. Guenther, J. M. Rothberg, and H. Karch. Prospective genomic characterization of the German enterohemorrhagic Escherichia coli O104:H4 outbreak by rapid next generation sequencing technology. *PLoS ONE*, 6(7):e22751, 2011.

[88] F. Meng, R. Henson, M. Lang, H. Wehbe, S. Maheshwari, J. T. Mendell, J. Jiang, T. D. Schmittgen, and T. Patel. Involvement of human micro-rna in growth and response to chemotherapy in human cholangiocarcinoma cell lines. *Gastroenterology*, 130(7):2113–2129, Jun 2006.

[89] M. L. Metzker. Sequencing technologies - the next generation. *Nat Rev Genet*, 11(1):31–46, Jan 2010.

[90] J. R. Miller, S. Koren, and G. Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 95(6):315–327, Jun 2010.

[91] A. E. Minoche, J. C. Dohm, and H. Himmelbauer. Evaluation of genomic high-throughput sequencing data generated on Illumina HiSeq and genome analyzer systems. *Genome Biol.*, 12(11):R112, 2011.

[92] F. Mitelman, B. Johansson, and F. Mertens. The impact of translocations and gene fusions on cancer causation. *Nat Rev Cancer*, 7(4):233–245, Apr 2007.

[93] J. G. Mulle, V. C. Patel, S. T. Warren, M. R. Hegde, D. J. Cutler, and M. E. Zwick. Empirical evaluation of oligonucleotide probe selection for DNA microarrays. *PLoS ONE*, 5(3):e9921, 2010.

[94] D. M. Muzny, M. N. Bainbridge, K. Chang, H. H. Dinh, J. A. Drummond, G. Fowler, C. L. Kovar, L. R. Lewis, M. B. Morgan, I. F. Newsham, J. G. Reid, J. Santibanez, E. Shinbrot, L. R. Trevino, Y. Q. Wu, M. Wang, P. Gunaratne, L. A. Donehower, C. J. Creighton, D. A. Wheeler, R. A. Gibbs, M. S. Lawrence, D. Voet, R. Jing, K. Cibulskis, A. Sivachenko, P. Stojanov, A. McKenna, E. S. Lander, S. Gabriel, G. Getz, L. Ding, R. S. Fulton, D. C. Koboldt, T. Wylie, J. Walker, D. J. Dooling, L. Fulton, K. D. Delehaunty, C. C. Fronick, R. Demeter, E. R. Mardis, R. K. Wilson, A. Chu, H. J. Chun, A. J. Mungall, E. Pleasance, A. Robertson, D. Stoll, M. Balasundaram, I. Birol, Y. S. Butterfield, E. Chuah, R. J. Coope, N. Dhalla, R. Guin, C. Hirst, M. Hirst, R. A. Holt, D. Lee, H. I. Li, M. Mayo, R. A. Moore, J. E. Schein, J. R. Slobodan, A. Tam, N. Thiessen, R. Varhol, T. Zeng, Y. Zhao, S. J. Jones, M. A. Marra, A. J. Bass, A. H. Ramos, G. Saksena, A. D. Cherniack, S. E. Schumacher, B. Tabak, S. L. Carter, N. H. Pho, H. Nguyen, R. C. Onofrio, A. Crenshaw, K. Ardlie, R. Beroukhim, W. Winckler, G. Getz, M. Meyerson, A. Protopopov, J. Zhang, A. Hadjipanayis, E. Lee, R. Xi, L. Yang, X. Ren, H. Zhang, N. Sathiamoorthy, S. Shukla, P. C. Chen, P. Haseley, Y. Xiao, S. Lee, J. Seidman, L. Chin, P. J. Park, R. Kucherlapati, J. T. Auman, K. A. Hoadley, Y. Du, M. D. Wilkerson, Y. Shi, C. Liquori, S. Meng, L. Li, Y. J. Turman, M. D. Topal, D. Tan, S. Waring, E. Buda, J. Walsh, C. D. Jones, P. A. Mieczkowski, D. Singh, J. Wu, A. Gulabani, P. Dolina, T. Bodenheimer, A. P. Hoyle, J. V. Simons, M. Soloway, L. E. Mose, S. R. Jefferys, S. Balu, B. D. O'Connor, J. F. Prins, D. Y. Chiang, D. Hayes, C. M. Perou, T. Hinoue, D. J. Weisenberger, D. T. Maglinte, F. Pan, B. P. Berman, D. J. Van Den Berg, H. Shen, T. Triche, S. B. Baylin, P. W. Laird, G. Getz, M. Noble, D. Voet, G. Saksena, N. Gehlenborg, D. DiCara, J. Zhang, H. Zhang, C. J. Wu, S. Y. Liu, S. Shukla, M. S. Lawrence, L. Zhou, A. Sivachenko, P. Lin, P. Stojanov, R. Jing, R. W. Park, M. D. Nazaire, J. Robinson, H. Thorvaldsdottir, J. Mesirov, P. J. Park, L. Chin, V. Thorsson, S. M. Reynolds, B. Bernard, R. Kreisberg, J. Lin, L. Iype, R. Bressler, T. Erkkila, M. Gundapuneni, Y. Liu, A. Norberg, T. Robinson, D. Yang, W. Zhang, I. Shmulevich, J. J. de Ronde, N. Schultz, E. Cerami, G. Ciriello, A. P. Goldberg, B. Gross, A. Jacobsen, J. Gao, B. Kaczkowski, R. Sinha, B. Aksoy, Y. Antipin, B. Reva, R. Shen, B. S. Taylor, T. A. Chan, M. Ladanyi, C. Sander, R. Akbani, N. Zhang, B. M. Broom, T. Casasent, A. Unruh, C. Wakefield, S. R. Hamilton, R. Cason, K. A. Baggerly, J. N. Weinstein, D. Haussler, C. C. Benz, J. M. Stuart, S. C. Benz, J. Sanborn, C. J. Vaske, J. Zhu, C. Szeto, G. K. Scott, C. Yau, S. Ng, T. Goldstein, K. Ellrott, E. Collisson, A. E. Cozen, D. Zerbino, C. Wilks, B. Craft, P. Spellman, R. Penny,

T. Shelton, M. Hatfield, S. Morris, P. Yena, C. Shelton, M. Sherman, J. Paulauskis, J. M. Gastier-Foster, J. Bowen, N. C. Ramirez, A. Black, R. Pyatt, L. Wise, P. White, M. Bertagnolli, J. Brown, T. A. Chan, G. C. Chu, C. Czerwinski, F. Denstman, R. Dhir, A. Dorner, C. S. Fuchs, J. G. Guillem, M. Iacocca, H. Juhl, A. Kaufman, B. Kohl, X. Van Le, M. C. Mariano, E. N. Medina, M. Meyers, G. M. Nash, P. B. Paty, N. Petrelli, B. Rabeno, W. G. Richards, D. Solit, P. Swanson, L. Temple, J. E. Tepper, R. Thorp, E. Vakiani, M. R. Weiser, J. E. Willis, G. Witkin, Z. Zeng, M. J. Zinner, C. Zornig, M. A. Jensen, R. Sfeir, A. B. Kahn, A. L. Chu, P. Kothiyal, Z. Wang, E. E. Snyder, J. Pontius, T. D. Pihl, B. Ayala, M. Backus, J. Walton, J. Whitmore, J. Baboud, D. L. Berton, M. C. Nicholls, D. Srinivasan, R. Raman, S. Girshik, P. A. Kigonya, S. Alonso, R. N. Sanbhadti, S. P. Barletta, J. M. Greene, D. A. Pot, K. R. Shaw, L. A. Dillon, K. Buetow, T. Davidsen, J. A. Demchok, G. Eley, M. Ferguson, P. Fielding, C. Schaefer, M. Sheth, L. Yang, M. S. Guyer, B. A. Ozenberger, J. D. Palchik, J. Peterson, H. J. Sofia, and E. Thomson. Comprehensive molecular characterization of human colon and rectal cancer. *Nature*, 487(7407):330–337, Jul 2012.

[95] E. W. Myers. Toward simplifying and accurately formulating fragment assembly. *J Comput Biol*, 2(2):275–290, 1995.

[96] E. W. Myers. The fragment assembly string graph. *Bioinformatics*, 21 Suppl 2:ii79–ii85, Sep 2005.

[97] K. Nakabayashi, S. Makino, S. Minagawa, A. C. Smith, J. S. Bamforth, P. Stanier, M. Preece, L. Parker-Katiraee, T. Paton, M. Oshimura, P. Mill, Y. Yoshikawa, C. C. Hui, D. Monk, G. E. Moore, and S. W. Scherer. Genomic imprinting of ppp1r9a encoding neurabin i in skeletal muscle and extra-embryonic tissues. *J Med Genet*, 41(8):601–608, Aug 2004.

[98] O. Nanopore. Oxford nanopore introduces dna 'strand sequencing' on the high-throughput gridion platform and presents minion, a sequencer the size of a usb memory stick. 2012.

[99] N. Navin, J. Kendall, J. Troge, P. Andrews, L. Rodgers, J. McIndoo, K. Cook, A. Stepansky, D. Levy, D. Esposito, L. Muthuswamy, A. Krasnitz, W. R. McCombie, J. Hicks, and M. Wigler. Tumour evolution inferred by single-cell sequencing. *Nature*, 472(7341):90–94, Apr 2011.

[100] N. Navin, A. Krasnitz, L. Rodgers, K. Cook, J. Meth, J. Kendall, M. Riggs, Y. Eberling, J. Troge, V. Grubor, D. Levy, P. Lundin, S. Maner, A. Zetterberg, J. Hicks, and M. Wigler. Inferring tumor progression from genomic heterogeneity. *Genome Res.*, 20(1):68–80, Jan 2010.

[101] B. M. Neale, Y. Kou, L. Liu, A. Ma'ayan, K. E. Samocha, A. Sabo, C. F. Lin, C. Stevens, L. S. Wang, V. Makarov, P. Polak, S. Yoon, J. Maguire, E. L. Crawford, N. G. Campbell, E. T. Geller, O. Valladares, C. Schafer, H. Liu, T. Zhao, G. Cai, J. Lihm, R. Dannenfelser, O. Jabado, Z. Peralta, U. Nagaswamy, D. Muzny, J. G. Reid, I. Newsham, Y. Wu, L. Lewis, Y. Han, B. F. Voight, E. Lim, E. Rossin, A. Kirby, J. Flannick, M. Fromer, K. Shakir, T. Fennell, K. Garimella, E. Banks, R. Poplin,

S. Gabriel, M. DePristo, J. R. Wimbish, B. E. Boone, S. E. Levy, C. Betancur, S. Sunyaev, E. Boerwinkle, J. D. Buxbaum, E. H. Cook, B. Devlin, R. A. Gibbs, K. Roeder, G. D. Schellenberg, J. S. Sutcliffe, and M. J. Daly. Patterns and rates of exonic de novo mutations in autism spectrum disorders. *Nature*, 485(7397):242–245, May 2012.

[102] T. P. Niedringhaus, D. Milanova, M. B. Kerby, M. P. Snyder, and A. E. Barron. Landscape of next-generation sequencing technologies. *Anal. Chem.*, 83(12):4327–4341, Jun 2011.

[103] S. Nik-Zainal, L. B. Alexandrov, D. C. Wedge, P. Van Loo, C. D. Greenman, K. Raine, D. Jones, J. Hinton, J. Marshall, L. A. Stebbings, A. Menzies, S. Martin, K. Leung, L. Chen, C. Leroy, M. Ramakrishna, R. Rance, K. W. Lau, L. J. Mudie, I. Varela, D. J. McBride, G. R. Bignell, S. L. Cooke, A. Shlien, J. Gamble, I. Whitmore, M. Maddison, P. S. Tarpey, H. R. Davies, E. Papaemmanuil, P. J. Stephens, S. McLaren, A. P. Butler, J. W. Teague, G. Jonsson, J. E. Garber, D. Silver, P. Miron, A. Fatima, S. Boyault, A. Langerod, A. Tutt, J. W. Martens, S. A. Aparicio, A. Borg, A. V. Salomon, G. Thomas, A. L. Borresen-Dale, A. L. Richardson, M. S. Neuberger, P. A. Futreal, P. J. Campbell, and M. R. Stratton. Mutational processes molding the genomes of 21 breast cancers. *Cell*, 149(5):979–993, May 2012.

[104] L. Oesper, A. Ritz, S. J. Aerni, R. Drebin, and B. J. Raphael. Reconstructing cancer genomes from paired-end sequencing data. *BMC Bioinformatics*, 13 Suppl 6:S10, 2012.

[105] A. B. Olshen, E. S. Venkatraman, R. Lucito, and M. Wigler. Circular binary segmentation for the analysis of array-based dna copy number data. *Biostatistics*, 5(4):557–572, Oct 2004.

[106] B. J. O'Roak, P. Deriziotis, C. Lee, L. Vives, J. J. Schwartz, S. Girirajan, E. Karakoc, A. P. Mackenzie, S. B. Ng, C. Baker, M. J. Rieder, D. A. Nickerson, R. Bernier, S. E. Fisher, J. Shendure, and E. E. Eichler. Exome sequencing in sporadic autism spectrum disorders identifies severe de novo mutations. *Nat. Genet.*, 43(6):585–589, Jun 2011.

[107] J. M. Perkel. Sanger who? sequencing the next generation. *Science Magazine*, 2009.

[108] B. A. Peters, B. G. Kermani, A. B. Sparks, O. Alferov, P. Hong, A. Alexeev, Y. Jiang, F. Dahl, Y. T. Tang, J. Haas, K. Robasky, A. W. Zaranek, J. H. Lee, M. P. Ball, J. E. Peterson, H. Perazich, G. Yeung, J. Liu, L. Chen, M. I. Kennemer, K. Pothuraju, K. Konvicka, M. Tsoupko-Sitnikov, K. P. Pant, J. C. Ebert, G. B. Nilsen, J. Baccash, A. L. Halpern, G. M. Church, and R. Drmanac. Accurate whole-genome sequencing and haplotyping from 10 to 20 human cells. *Nature*, 487(7406):190–195, Jul 2012.

[109] P. A. Pevzner, P. A. Pevzner, H. Tang, and G. Tesler. De novo repeat classification and fragment assembly. *Genome Res*, 14(9):1786–1796, Sep 2004.

[110] P. A. Pevzner and H. Tang. Fragment assembly with double-barreled data. *Bioinformatics*, 17 Suppl 1:S225–S233, 2001.

[111] P. A. Pevzner, H. Tang, and M. S. Waterman. An eulerian path approach to dna fragment assembly. *Proc Natl Acad Sci U S A*, 98(17):9748–9753, Aug 2001.

[112] F. Picard, S. Robin, M. Lavielle, C. Vaisse, and J.-J. Daudin. A statistical approach for array cgh data analysis. *BMC Bioinformatics*, 6:27, 2005.

[113] D. Pinkel and D. G. Albertson. Array comparative genomic hybridization and its applications in cancer. *Nat Genet*, 37 Suppl:S11–S17, Jun 2005.

[114] D. Pinkel, R. Segraves, D. Sudar, S. Clark, I. Poole, D. Kowbel, C. Collins, W. L. Kuo, C. Chen, Y. Zhai, S. H. Dairkee, B. M. Ljung, J. W. Gray, and D. G. Albertson. High resolution analysis of dna copy number variation using comparative genomic hybridization to microarrays. *Nat Genet*, 20(2):207–211, Oct 1998.

[115] M. Pop. Genome assembly reborn: recent computational challenges. *Brief Bioinform*, 10(4):354–366, Jul 2009.

[116] D. Pushkarev, N. F. Neff, and S. R. Quake. Single-molecule sequencing of an individual human genome. *Nat. Biotechnol.*, 27(9):847–850, Sep 2009.

[117] Q. Qiu, G. Zhang, T. Ma, W. Qian, J. Wang, Z. Ye, C. Cao, Q. Hu, J. Kim, D. M. Larkin, L. Auvil, B. Capitanu, J. Ma, H. A. Lewin, X. Qian, Y. Lang, R. Zhou, L. Wang, K. Wang, J. Xia, S. Liao, S. Pan, X. Lu, H. Hou, Y. Wang, X. Zang, Y. Yin, H. Ma, J. Zhang, Z. Wang, Y. Zhang, D. Zhang, T. Yonezawa, M. Hasegawa, Y. Zhong, W. Liu, Y. Zhang, Z. Huang, S. Zhang, R. Long, H. Yang, J. Wang, J. A. Lenstra, D. N. Cooper, Y. Wu, J. Wang, P. Shi, J. Wang, and J. Liu. The yak genome and adaptation to life at high altitude. *Nat. Genet.*, 44(8):946–949, 2012.

[118] M. Quail, M. E. Smith, P. Coupland, T. D. Otto, S. R. Harris, T. R. Connor, A. Bertoni, H. P. Swerdlow, and Y. Gu. A tale of three next generation sequencing platforms: comparison of Ion torrent, pacific biosciences and illumina MiSeq sequencers. *BMC Genomics*, 13(1):341, Jul 2012.

[119] A. R. Quinlan, R. A. Clark, S. Sokolova, M. L. Leibowitz, Y. Zhang, M. E. Hurles, J. C. Mell, and I. M. Hall. Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome. *Genome Res*, 20(5):623–635, May 2010.

[120] B. J. Raphael, S. Volik, P. Yu, C. Wu, G. Huang, E. V. Linardopoulou, B. J. Trask, F. Waldman, J. Costello, K. J. Pienta, G. B. Mills, K. Bajsarowicz, Y. Kobayashi, S. Sridharan, P. L. Paris, Q. Tao, S. J. Aerni, R. P. Brown, A. Bashir, J. W. Gray, J.-F. Cheng, P. de Jong, M. Nefedov, T. Ried, H. M. Padilla-Nash, and C. C. Collins. A sequence-based survey of the complex structural organization of tumor genomes. *Genome Biol*, 9(3):R59, 2008.

[121] T. Rausch, S. Koren, G. Denisov, D. Weese, A.-K. Emde, A. Dring, and K. Reinert. A consistency-based consensus algorithm for de novo and reference-guided sequence assembly of short reads. *Bioinformatics*, 25(9):1118–1124, May 2009.

[122] T. Rausch, T. Zichner, A. Schlattl, A. M. Stutz, V. Benes, and J. O. Korbel. DELLY: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, 28(18):i333–i339, Sep 2012.

[123] R. Redon, S. Ishikawa, K. R. Fitch, L. Feuk, G. H. Perry, T. D. Andrews, H. Fiegler, M. H. Shapero, A. R. Carson, W. Chen, E. K. Cho, S. Dallaire, J. L. Freeman, J. R. Gonzlez, M. Gratacs, J. Huang, D. Kalaitzopoulos, D. Komura, J. R. MacDonald, C. R. Marshall, R. Mei, L. Montgomery, K. Nishimura, K. Okamura, F. Shen, M. J. Somerville, J. Tchinda, A. Valsesia, C. Woodwark, F. Yang, J. Zhang, T. Zerjal, J. Zhang, L. Armengol, D. F. Conrad, X. Estivill, C. Tyler-Smith, N. P. Carter, H. Aburatani, C. Lee, K. W. Jones, S. W. Scherer, and M. E. Hurles. Global variation in copy number in the human genome. *Nature*, 444(7118):444–454, Nov 2006.

[124] A. Ritz, A. Bashir, and B. J. Raphael. Structural variation analysis with strobe reads. *Bioinformatics*, 26(10):1291–1298, May 2010.

[125] A. Ritz, P. L. Paris, M. M. Ittmann, C. Collins, and B. J. Raphael. Detection of recurrent rearrangement breakpoints from copy number data. *BMC Bioinformatics*, To Appear.

[126] F. Sanger, S. Nicklen, and A. R. Coulson. Dna sequencing with chain-terminating inhibitors. *Proc Natl Acad Sci U S A*, 74(12):5463–5467, Dec 1977.

[127] M. C. Schatz, A. L. Delcher, and S. L. Salzberg. Assembly of large genomes using second-generation sequencing. *Genome Res.*, 20(9):1165–1173, Sep 2010.

[128] S. W. Scherer, C. Lee, E. Birney, D. M. Altshuler, E. E. Eichler, N. P. Carter, M. E. Hurles, and L. Feuk. Challenges and standards in integrating surveys of structural variation. *Nat Genet*, 39(7 Suppl):S7–15, Jul 2007.

[129] S. C. Schuster, W. Miller, A. Ratan, L. P. Tomsho, B. Giardine, L. R. Kasson, R. S. Harris, D. C. Petersen, F. Zhao, J. Qi, C. Alkan, J. M. Kidd, Y. Sun, D. I. Drautz, P. Bouffard, D. M. Muzny, J. G. Reid, L. V. Nazareth, Q. Wang, R. Burhans, C. Riemer, N. E. Wittekindt, P. Moorjani, E. A. Tindall, C. G. Danko, W. S. Teo, A. M. Buboltz, Z. Zhang, Q. Ma, A. Oosthuysen, A. W. Steenkamp, H. Oostuisen, P. Venter, J. Gajewski, Y. Zhang, B. F. Pugh, K. D. Makova, A. Nekrutenko, E. R. Mardis, N. Patterson, T. H. Pringle, F. Chiaromonte, J. C. Mullikin, E. E. Eichler, R. C. Hardison, R. A. Gibbs, T. T. Harkins, and V. M. Hayes. Complete Khoisan and Bantu genomes from southern Africa. *Nature*, 463(7283):943–947, Feb 2010.

[130] A. J. Sharp, Z. Cheng, and E. E. Eichler. Structural variation of the human genome. *Annu Rev Genomics Hum Genet*, 7:407–442, 2006.

[131] J. Shendure and H. Ji. Next-generation DNA sequencing. *Nat. Biotechnol.*, 26(10):1135–1145, Oct 2008.

[132] J. T. Simpson, K. Wong, S. D. Jackman, J. E. Schein, S. J. M. Jones, and I. Birol. Abyss: a parallel assembler for short read sequence data. *Genome Res*, 19(6):1117–1123, Jun 2009.

[133] S. Sindi, E. Helman, A. Bashir, and B. J. Raphael. A geometric approach for classification and comparison of structural variants. *Bioinformatics*, 25(12):i222–i230, Jun 2009.

[134] S. S. Sindi, S. Onal, L. C. Peng, H. T. Wu, and B. J. Raphael. An integrative probabilistic model for identification of structural variation in sequencing data. *Genome Biol.*, 13(3):R22, 2012.

[135] D. D. Sommer, A. L. Delcher, S. L. Salzberg, and M. Pop. Minimus: a fast, lightweight genome assembler. *BMC Bioinformatics*, 8:64, 2007.

[136] P. J. Stephens, C. D. Greenman, B. Fu, F. Yang, G. R. Bignell, L. J. Mudie, E. D. Pleasance, K. W. Lau, D. Beare, L. A. Stebbings, S. McLaren, M.-L. Lin, D. J. McBride, I. Varela, S. Nik-Zainal, C. Leroy, M. Jia, A. Menzies, A. P. Butler, J. W. Teague, M. A. Quail, J. Burton, H. Swerdlow, N. P. Carter, L. A. Morsberger, C. Iacobuzio-Donahue, G. A. Follows, A. R. Green, A. M. Flanagan, M. R. Stratton, P. A. Futreal, and P. J. Campbell. Massive genomic rearrangement acquired in a single catastrophic event during cancer development. *Cell*, 144(1):27 – 40, 2011.

[137] T. Sun, N. Aceto, K. L. Meerbrey, J. D. Kessler, C. Zhou, I. Migliaccio, D. X. Nguyen, N. N. Pavlova, M. Botero, J. Huang, R. J. Bernardi, E. Schmitt, G. Hu, M. Z. Li, N. Dephoure, S. P. Gygi, M. Rao, C. J. Creighton, S. G. Hilsenbeck, C. A. Shaw, D. Muzny, R. A. Gibbs, D. A. Wheeler, C. K. Osborne, R. Schiff, M. Bentires-Alj, S. J. Elledge, and T. F. Westbrook. Activation of multiple proto-oncogenic tyrosine kinases in breast cancer via loss of the PTPN12 phosphatase. *Cell*, 144(5):703–718, Mar 2011.

[138] A. Sundquist, M. Ronaghi, H. Tang, P. Pevzner, and S. Batzoglou. Whole-genome sequencing and assembly with high-throughput, short-read technologies. *PLoS One*, 2(5):e484, 2007.

[139] S. A. Tomlins, D. R. Rhodes, S. Perner, S. M. Dhanasekaran, R. Mehra, X.-W. Sun, S. Varambally, X. Cao, J. Tchinda, R. Kuefer, C. Lee, J. E. Montie, R. B. Shah, K. J. Pienta, M. A. Rubin, and A. M. Chinnaiyan. Recurrent fusion of tmprss2 and ets transcription factor genes in prostate cancer. *Science*, 310(5748):644–648, Oct 2005.

[140] T. J. Treangen, D. D. Sommer, F. E. Angly, S. Koren, and M. Pop. Next generation sequence assembly with amos. *Curr Protoc Bioinformatics*, Chapter 11:Unit11.8, Mar 2011.

[141] S. Turajlic, S. J. Furney, M. B. Lambros, C. Mitsopoulos, I. Kozarewa, F. C. Geyer, A. Mackay, J. Hakas, M. Zvelebil, C. J. Lord, A. Ashworth, M. Thomas, G. Stamp, J. Larkin, J. S. Reis-Filho, and R. Marais. Whole genome sequencing of matched primary and metastatic acral melanomas. *Genome Res.*, 22(2):196–207, Feb 2012.

[142] S. Turner. Personal genomes (conference talk). Cold Spring Harbor Laboratory, 2009.

[143] E. Tuzun, A. J. Sharp, J. A. Bailey, R. Kaul, V. A. Morrison, L. M. Pertz, E. Haugen, H. Hayden, D. Albertson, D. Pinkel, M. V. Olson, and E. E. Eichler. Fine-scale structural variation of the human genome. *Nat Genet*, 37(7):727–732, Jul 2005.

[144] E. S. Venkatraman and A. B. Olshen. A faster circular binary segmentation algorithm for the analysis of array cgh data. *Bioinformatics*, 23(6):657–663, Mar 2007.

[145] V. Vladimirova, A. Waha, K. Lckerath, P. Pesheva, and R. Probstmeier. Runx2 is expressed in human glioma cells and mediates the expression of galectin-3. *J Neurosci Res*, 86(11):2450–2461, Aug 2008.

[146] B. Vogelstein and K. W. Kinzler. The multistep nature of cancer. *Trends Genet.*, 9(4):138–141, Apr 1993.

[147] S. Volik, B. J. Raphael, G. Huang, M. R. Stratton, G. Bignel, J. Murnane, J. H. Brebner, K. Bajsarowicz, P. L. Paris, Q. Tao, D. Kowbel, A. Lapuk, D. A. Shagin, I. A. Shagina, J. W. Gray, J.-F. Cheng, P. J. de Jong, P. Pevzner, and C. Collins. Decoding the fine-scale structure of a breast cancer genome and transcriptome. *Genome Res*, 16(3):394–404, Mar 2006.

[148] R. Walpole, R. Myers, S. Myers, and K. Ye. *Probability & Statistics for Engineers & Scientists*. Prentice-Hall, Upper Saddle River, NJ, 7 edition, 2002.

[149] T. Walsh, J. M. McClellan, S. E. McCarthy, A. M. Addington, S. B. Pierce, G. M. Cooper, A. S. Nord, M. Kusenda, D. Malhotra, A. Bhandari, S. M. Stray, C. F. Rippey, P. Roccanova, V. Makarov, B. Lakshmi, R. L. Findling, L. Sikich, T. Stromberg, B. Merriman, N. Gogtay, P. Butler, K. Eckstrand, L. Noory, P. Gochman, R. Long, Z. Chen, S. Davis, C. Baker, E. E. Eichler, P. S. Meltzer, S. F. Nelson, A. B. Singleton, M. K. Lee, J. L. Rapoport, M.-C. King, and J. Sebat. Rare structural variants disrupt multiple genes in neurodevelopmental pathways in schizophrenia. *Science*, 320(5875):539–543, Apr 2008.

[150] V. Walter, A. B. Nobel, and F. A. Wright. Dinamic: a method to identify recurrent dna copy number aberrations in tumors. *Bioinformatics*, 27(5):678–685, Mar 2011.

[151] R. L. Warren, G. G. Sutton, S. J. M. Jones, and R. A. Holt. Assembling millions of short dna sequences using ssake. *Bioinformatics*, 23(4):500–501, Feb 2007.

[152] J. L. Weber, D. David, J. Heil, Y. Fan, C. Zhao, and G. Marth. Human diallelic insertion/deletion polymorphisms. *Am J Hum Genet*, 71(4):854–862, Oct 2002.

[153] R. Xi, T.-M. Kim, and P. J. Park. Detecting structural variations in the human genome using next generation sequencing. *Brief Funct Genomics*, 9(5-6):405–415, Dec 2010.

[154] I. M. V. G. B. Yaglom. *Convex Figures*.

[155] K. Ye, M. H. Schulz, Q. Long, R. Apweiler, and Z. Ning. Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads. *Bioinformatics*, 25(21):2865–2871, Nov 2009.

[156] S. Yilmaz and A. K. Singh. Single cell genome sequencing. *Curr. Opin. Biotechnol.*, 23(3):437–443, Jun 2012.

[157] S. Yoon, Z. Xuan, V. Makarov, K. Ye, and J. Sebat. Sensitive and accurate detection of copy number variants using read depth of coverage. *Genome Res*, 19(9):1586–1592, Sep 2009.

[158] D. R. Zerbino. Using the velvet de novo assembler for short-read sequencing technologies. *Curr Protoc Bioinformatics*, Chapter 11:Unit 11.5, Sep 2010.

[159] D. R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Res*, 18(5):821–829, May 2008.

[160] D. R. Zerbino, G. K. McEwen, E. H. Margulies, and E. Birney. Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. *PLoS One*, 4(12):e8407, 2009.

[161] J. Zhang, L. Feuk, G. E. Duggan, R. Khaja, and S. W. Scherer. Development of bioinformatics resources for display and analysis of copy number and other structural variants in the human genome. *Cytogenet Genome Res*, 115(3-4):205–214, 2006.

[162] N. R. Zhang and D. O. Siegmund. A modified bayes information criterion with applications to the analysis of comparative genomic hybridization data. *Biometrics*, 63(1):22–32, Mar 2007.

[163] Q. Zhang, L. Ding, D. E. Larson, D. C. Koboldt, M. D. McLellan, K. Chen, X. Shi, A. Kraja, E. R. Mardis, R. K. Wilson, I. B. Borecki, and M. A. Province. Cmds: a population-based method for identifying recurrent dna copy number aberrations in cancer from high-resolution data. *Bioinformatics*, 26(4):464–469, Feb 2010.

[164] W. Zhang, J. Chen, Y. Yang, Y. Tang, J. Shang, and B. Shen. A practical comparison of de novo genome assembly software tools for next-generation sequencing technologies. *PLoS One*, 6(3):e17915, 2011.