nFuse: Discovery of complex genomic rearrangements in cancer using high-throughput sequencing

Andrew William McPherson, Chunxiao Wu, Alexander Wyatt, et al.

Genome Res. published online June 28, 2012
Access the most recent version at doi:10.1101/gr.136572.111

nFuse: Discovery of complex genomic rearrangements in cancer using high-throughput sequencing

Andrew McPherson [*1], Chunxiao Wu[2], Alexander Wyatt[2], Sohrab Shah[3], Colin Collins[2] and S. Cenk Sahinalp [†1]

[1]School of Computing Science, Simon Fraser University
[2]Vancouver Prostate Centre
[3]Department of Molecular Oncology, BC Cancer Research Centre

June 13, 2012

Abstract

Complex Genomic Rearrangements (CGRs) are emerging as a new feature of cancer genomes. CGRs are characterized by multiple genomic breakpoints, and thus have the potential to simultaneously affect multiple genes, fusing some genes and interrupting other genes.

Analysis of high-throughput whole genome shotgun sequencing (WGSS) is beginning to facilitate the discovery and characterization of CGRs, but further development of computational methods is required. We have developed an algorithmic method for identifying CGRs in WGSS data based on shortest alternating paths in breakpoint graphs. Aiming for a method with the highest possible sensitivity, we use breakpoint graphs built from all WGSS data, including sequences with ambiguous genomic origin. Since the majority of cell function is encoded by the transcriptome, we target our search to find CGRs that underlie fusion transcripts predicted from matched high-throughput cDNA sequencing (RNA-seq)

We have applied our method, nFuse, to the discovery of CGRs in publicly available data from the well-studied breast cancer cell line HCC1954 and primary prostate tumour sample 963. We first establish the sensitivity and specificity of the nFuse breakpoint prediction and scoring method using breakpoints previously discovered in HCC1954. We then validate 5 out of 6 CGRs in HCC1954 and 2 out of 2 CGRs in 963. We show examples of gene fusions that would be difficult to discover using methods that do not account for the existence of CGRs, including one important event that was missed in a previous study of the HCC1954 genome. Finally, we illustrate how CGRs may be used to infer the gene expression history of a tumour.

[*]email: andrew.mcpherson@gmail.com
[†]email: cenk@sfu.ca

Introduction

Cancer is a genomic disease characterized by unregulated cell growth resulting from acquired or inherited DNA changes. Genome rearrangements are an important class of DNA changes, known to disrupt the activity of tumour suppressor genes and promote increased activity of oncogenes. Genome rearrangements are also known to create fusion genes: novel oncogenes formed when a rearrangement juxtaposes two or more existing genes. Fusion genes are the defining molecular feature of many cancers and represent potential drug targets in those cancers. A classic example is the *BCR-ABL1* gene fusion present in 95% of chronic myelogenous leukemia patients, and targeted by the drug iminitib.

The molecular mechanisms that cause somatic genome rearrangements are still the focus of investigation. Double stranded DNA breaks followed by a 'joining event' are known to result in a *simple genomic rearrangement* consisting of a single breakpoint, where a breakpoint is defined as a pair of genomic locations that are distant in the normal genome, but adjacent in the tumour genome. A breakpoint can be considered as the most basic unit of rearrangement. Examples of processes that generate single breakpoints include breakage-fusion-bridge cycles, nonhomologous end joining and homologous recombination-mediated repair (Bignell et al., 2007).

Recently discovered are *complex genomic rearrangements* (CGRs), rearrangements comprised of multiple breakpoints with a specific structure. In prostate cancer, for example, Berger et al. (2011) discovered *closed chains of breakage and rejoining* (CCBR). Berger et al. (2011) suggested that CCBR potentially occurs when distant chromosomal regions are spatially co-localized in the nucleus, possibly because they have been recruited by the same transcriptional factory. Importantly, they showed that biologically relevant gene fusions, such as TMPRSS2-ERG, were created by CCBR events. CCBRs are *balanced* rearrangements: they result in little or no loss of genomic material. It has been proposed that balanced rearrangements are more likely to produce functional gene fusions (Mitelman et al., 2007).

Other cancers, exhibit an entirely different type of CGR produced by a shattering of chromosomal regions, followed by a reassembly from the resulting fragments (Stephens et al., 2011). As a result, some breakpoints between large chromosomal segments contain additional smaller fragments (*genomic shards*) interposed at the breakpoint (Bignell et al., 2007). These genomic shards originate from other regions affected by the catastrophe, typically at the boundaries of deleted regions (Bignell et al., 2007). Breakpoints with small (∼500bp) genomic shards interposed at the breakpoint are termed *complex* and have been identified previously in breast cancer (Stephens et al., 2009). Breakpoints with larger fragments of other genes interposed at the breakpoint have the potential to create *poly-fusions*, fusion genes comprised of 3 or more separate genes. Both complex breakpoints and poly-fusions are rearrangements composed of two or more simple breakpoints, and identification of all breakpoints is required to discover the fusion.

High-throughput paired-end Whole Genome Shotgun Sequencing (WGSS) is currently the most efficient method of identifying breakpoints in tumour genomes. Briefly, WGSS can be used to sequence the ends of short fragments of DNA produced by fragmentation of a tumour genome. The pairs of end sequences (paired-end reads, or simply reads) can then be mapped back to a healthy reference genome sequence. Distantly mapping reads or reads that map with unexpected orientation can then be used to predict breakpoints. WGSS,

however, presents many unique challenges compared to earlier technologies. The presence of repeated regions in the genome and short WGSS read lengths complicate the problem of unambiguously identifying the origin of some WGSS reads. Furthermore, sequencing errors lead to some proportion of false reads. Both of these problems are magnified due to the huge size of WGSS datasets. Finally, aneuploidy, tumour heterogeneity and cellularity have the combined affect of diluting the sequence signal of breakpoints, even in high coverage WGSS datasets. Nevertheless, solutions now exist for accurately predicting breakpoints from WGSS (Hormozdiari et al., 2009; Chen et al., 2009), though a true account of false negative rates remains elusive.

Given the ability to predict breakpoints in WGSS, an important question is how to infer genome structure from these breakpoints, and potentially reconstruct chromosomal architectures. In a recent paper, Greenman et al. (2011) propose methods for reconstructing 'digital karyotypes' from copy number and breakpoint predictions. Their method requires precise breakpoint predictions, and could not guarantee a unique solution for a reasonably complex genome. Previous to Greenman et al. (2011), efforts to reconstruct tumour genomes relied on low resolution data such as FISH and BAC sequencing (Raphael and Pevzner, 2004; Raphael et al., 2003; Ozery-Flato and Shamir, 2009). These methods may be sufficiently sensitive to reconstruct large scale rearrangements, however they will likely miss complex focal rearrangements.

In this study, we propose a method for reconstructing CGRs from WGSS data. Crucial to the problem of identifying CGRs is the missing data problem: identification of a CGR relies on the identification of all n breakpoints in the CGR. Therefore, the basis for our approach is a high sensitivity method for predicting breakpoints. However, WGSS read alignment data contains a significant amount of noise, and this noise will produce false positive predictions, especially with a method that prioritizes sensitivity. Thus we calculate a probability for each breakpoint that reflects our belief in its existence. Like the aforementioned studies, we identify CGRs using *breakpoint graphs* (Pevzner, 2000). We incorporate the breakpoint probability into the graph, and use that probability to guide our search for high probability structures representing potential CGRs. We prioritize our search for CGRs based on fusion transcript predictions from matched high-throughput cDNA sequencing (RNA-seq), thereby using effect on the transcriptome as an indicator of potential functional significance.

We have applied our method, nFuse, to publicly available WGSS and RNA-seq data for the well characterized breast cancer cell line HCC1954. We show that we are able to rediscover a significant proportion of previously discovered breakpoints. Furthermore, we show that the breakpoint probability we calculate accurately separates the previously discovered breakpoints from a background of predominantly false positive predictions. Using Long-Range PCR (LR-PCR), we validated 5 out of 6 poly-fusions predicted by nFuse for HCC1954. We have also applied nFuse to WGSS and RNA-seq data generated from primary human prostate cancer sample 963 (Wu et al., 2012). Using a CCBR discovered in 963, we illustrate how CCBRs can be used to infer the gene expression history of a tumour. Finally, we present an example of a CCBR with a complex breakpoint discovered in 963, providing a link between CCBR and complex breakpoints.

Methods

Complex rearrangement discovery using breakpoint graphs

Complex rearrangements involve two or more breakpoints, such that the set of breakpoints elicit a specific structure. To identify complex rearrangements, we employ a construct called the *breakpoint graph* (Pevzner, 2000). The complex rearrangements we are interested in discovering naturally arise as features of the breakpoint graph. Unlike previous breakpoint graph approaches, the breakpoint graph we construct includes a measure of the uncertainty inherent in breakpoint predictions produced from WGSS data. Our algorithms then seek to identify CGRs more likely to be real by searching for the higher probability structures in the breakpoint graph.

Of crucial importance is the effect of missing data on our ability to predict CGRs. For a CGR composed of n breakpoints, failing to predict any one of those n breakpoints will result in a failure to identify the CGR. To mitigate this problem we seek to include in the breakpoint graph *all* reasonable breakpoint predictions, including those nominated by reads with ambiguous genomic origin. Thus the breakpoint graph we construct will contain a large amount of noise, and the majority of breakpoints are expected to be false positives. A real but low probability breakpoint may then be identified as part of a CGR, providing the probabilities of the CGR's other breakpoints are sufficiently high. By contrast, removing low probability breakpoints before building the breakpoint graph would also remove the aforementioned real breakpoint, making it impossible to identify the CGR.

nFuse seeks to identify two types of CGRs: Closed Chains of Breakage and Rejoining (CCBR) (Berger et al., 2011), and poly-fusions/complex breakpoints. We emphasize here that these two types of CGRs are very different types of events, unified by breakpoint graphs as a common computational representation. We introduce the concept of the breakpoint graph by first focusing on poly-fusions and complex breakpoints, after which we describe CCBRs and their breakpoint graph representation.

Breakpoint graph structure

A breakpoint is an adjacency in one genome that does not exist in another genome. In the context of cancer genomics, we are interested in identifying adjacencies in the tumour genome not found in the normal (or reference) genome. Such unexpected adjacencies are evidence of somatic rearrangement, and may have important implications for tumour biology. For instance, an unexpected adjacency between the 5' exons of gene A and the 3' exons of gene B may represent an A-B fusion gene that drives proliferation of a tumour.

The breakpoint graph is a representation of a set of unexpected adjacencies, or breakpoints. We use the breakpoint graph to represent the set of breakpoints identified in a tumour genome, that are not in the reference genome. The graph is defined on a set of vertices representing the set of nucleotides that are adjacent in the reference but not in the tumour. The graph contains two types of edges, breakpoint edges and adjacency edges. Breakpoint edges represent adjacencies in the tumour, while adjacency edges represent a putatively contiguous region of the reference genome not interrupted by a breakpoint. Note that for identification of CCBRs, we generalize adjacency edges as described below.

Consider the following example illustrated in Figure 1. Let A_1 and A_2 be adjacent nucleotides in reference chromosome A, B_1 and B_2 be adjacent nucleotides in reference chromosome B, and suppose we identify an A_1, B_2 breakpoint (Figure 1a). The graph for the A_1, B_2 breakpoint contains vertices for A_1, A_2, B_1 and B_2, in addition to the *breakpoint edge* (A_1, B_2) (Figure 1b). Now consider an additional B_3, C_1 breakpoint between chromosomes B and C (Figure 1c). In addition to the (B_3, C_1) breakpoint edge, the graph also contains a (B_2, B_3) adjacency edge representing a putatively contiguous region in the tumour between nucleotides B_2 and B_3 (Figure 1d). Finally consider a fourth breakpoint between chromosomes A and B (Figure 1e) represented by an (A_3, B_5) breakpoint edge (Figure 1f). The breakpoint graph will also contain a (B_2, B_5) adjacency edge representing the possibility that the (B_3, C_1) does not exist in a putative tumour chromosome that contains both the (A_1, B_2) and (B_5, A_3) breakpoints. Thus each breakpoint will be considered *optional* in our realization of the breakpoint graph. To reflect this, we define adjacency edges as follows. Let $X_{\text{left}}, X_{\text{right}}$ be two nucleotides adjacent in the reference with a breakpoint edge incident on X_{left}. We add an adjacency edge from X_{left} to every upstream *right* vertex. Similarly, if a breakpoint edge is incident on X_{right}, we add an adjacency edge from X_{right} to every downstream *left* vertex.

Poly-fusions and complex breakpoints

A key feature of the breakpoint graph is that every alternating path represents a putative tumour chromosome. Poly-fusions and complex breakpoints are subsequences of tumour chromosomes, and as such will be represented as alternating paths given successful identification of all relevant breakpoints. As an example, consider a fusion between gene X on chromosome A and gene Y on chromosome C, for which a fragment of chromosome B is interposed at the breakpoint (Figure 1g). In the breakpoint graph, the complex breakpoint will be represented as an alternating path of length 5 between vertices representing the 5' end of gene X and the 3' end of gene Y (Figure 1h). In general, a poly-fusion or complex breakpoint involving n loci will be represented in the breakpoint graph as an alternating path of length $2n - 3$.

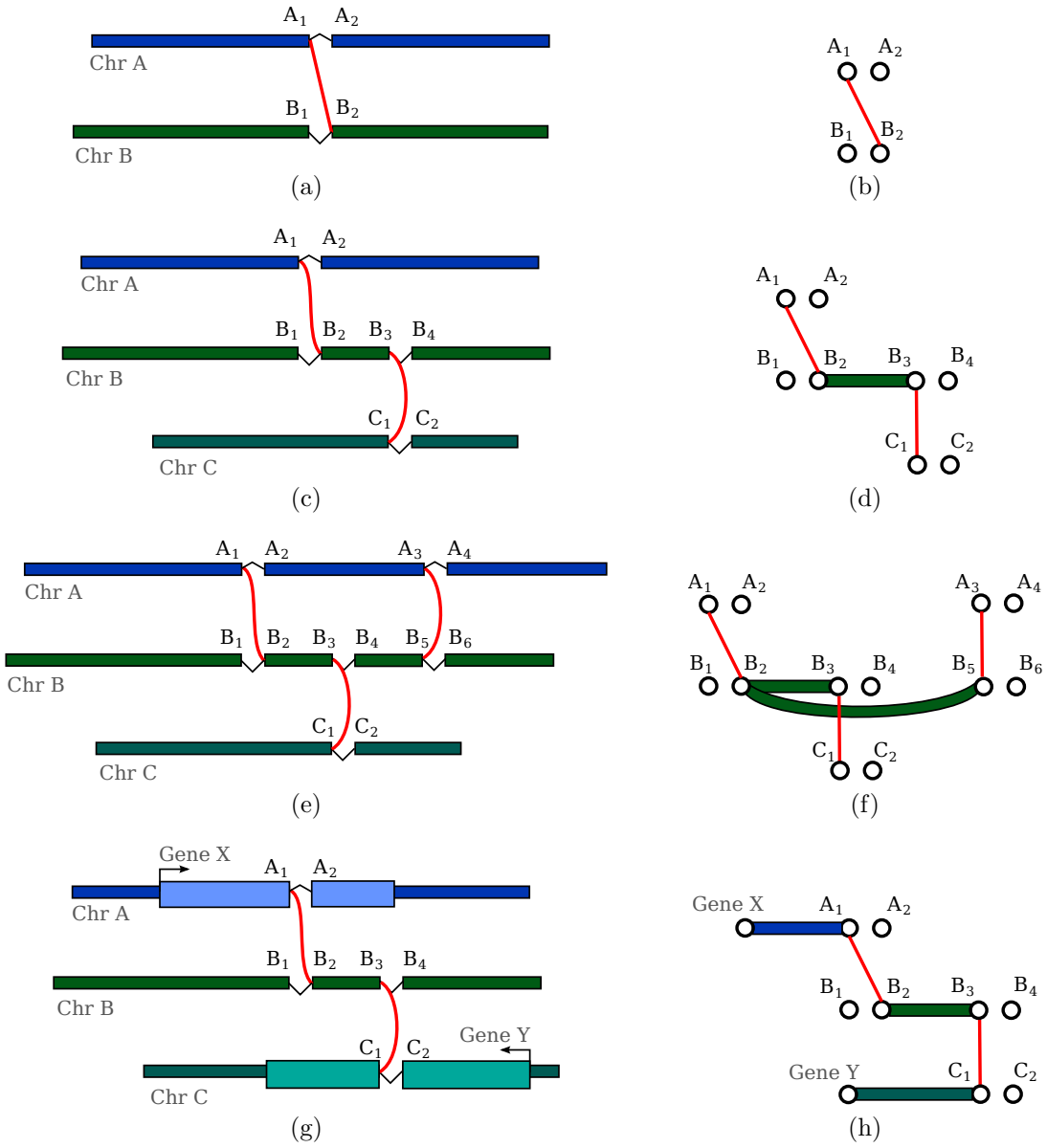


Figure 1: Breakpoint graphs and poly-fusions. (a) A breakpoint as an unexpected adjacency. (b) The breakpoint graph for a single breakpoint showing a breakpoint edge (red). (c) Two breakpoints on chromosomes A, B, and C. (d) The breakpoint graph for the two breakpoints showing 2 breakpoint edges (red) and an adjacency edge (green). (e) Three breakpoints. (f) The breakpoint graph for the three breakpoints showing a (B_2, B_5) adjacency edge that encodes the optional nature of breakpoint (B_3, C_1). (g) Breakpoints for a X-Y gene fusion with a complex breakpoint. (h) The breakpoint graph for the complex breakpoint showing an alternating path between X and Y.

Closed chains of breakage and rejoining (CCBR)

Closed chains of breakage and rejoining (CCBR) can be thought of as a generalization of a reciprocal translocation to $n > 2$ loci. For a reciprocal translocation, 2 loci are broken and the broken ends are swapped and rejoined. A 3 loci CCBR involves the breakage, permutation and rejoining of 3 loci. An example 3 loci CCBR event would be the transformation of chromosomes A, B and C into tumour chromosomes A-B, B-C and C-A (Figure 2a).

In the ideal case, no chromosomal material will be lost in the exchange (Figure 2a). As shown in this work and previously (Berger et al., 2011), many instances of chromosomal breakage and rejoining involve the loss or gain of chromosomal material. As a result, the breakpoints at broken and rejoined loci may be separated by an unknown distance. Figure 2b depicts a more realistic example involving chromosomes A, B and C. In this example, the CCBR has resulted in a loss of small sections of chromosomes A and C. In addition, the A-B and B-C tumour chromosomes created by the CCBR both include copies of a segment of chromosome B, resulting in a gain of that segment. Of crucial importance, any loss or gain caused by a CCBR will not necessarily be represented by additional breakpoints in the breakpoint graph. Nevertheless, the breakpoint graph, properly defined, will yield CCBRs as a specific type of subgraph.

To identify CCBRs we augment the previously defined breakpoint graph with additional edges. Call the previously defined adjacency edges as *gain* adjacency edges. Define additional adjacency edges called *loss* adjacency edges as follows. Let $X_{\text{left}}, X_{\text{right}}$ be two nucleotides adjacent in the reference with a breakpoint edge incident on X_{left}. Add loss adjacency edges from X_{left} to every downstream *right* vertex. Similarly, if a breakpoint edge is incident on X_{right}, add loss adjacency edges from X_{right} to every upstream *left* vertex. A n-loci CCBR in the resulting graph will be represented by an alternating cycle of length $2n$. Figure 2c shows the breakpoint graph for the CCBR in Figure 2b. The breakpoint edges, loss edges (A_1, A_4) and (C_1, C_4), and gain edge (B_2, B_3) together form an alternating 6-cycle. Note that an alternative explanation for the breakpoints in Figure 2b is a reciprocal translocation between chromosomes A and C, with a complex breakpoint for the A-C chromosome involving a shard of chromosome B. We will explore this ambiguity further when discussing the results for tumour sample 963.

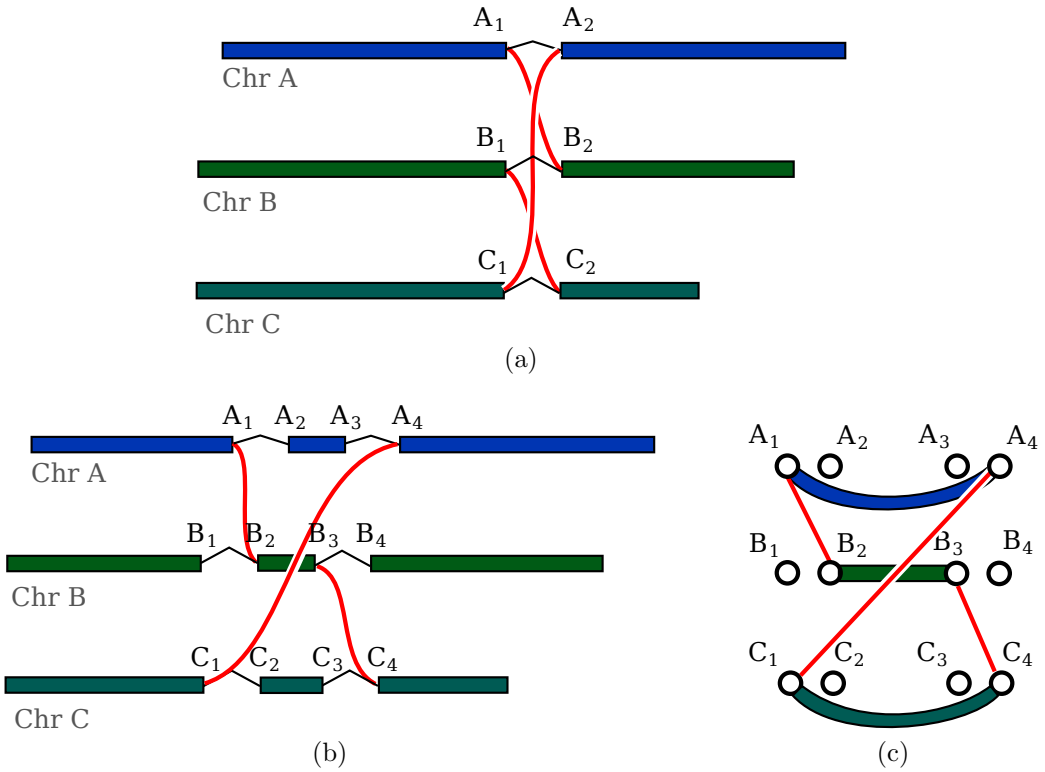


Figure 2: Closed Chains of Breakage and Rejoining (CCBR) (a) In an idealized version of a CCBR, no chromosomal material is lost or gained. (b) Actual CCBRs may involve small loss or gain of chromosomal material. For instance, the $A_2 \rightarrow A_3$ and $C_2 \rightarrow C_3$ sections of chromosomes A and C appear to have been lost, and the $B_2 \rightarrow B_3$ section of chromosome B appears to have been duplicated. (c) The breakpoint graph for the CCBR in figure 2b showing (A_1, A_4) and (C_1, C_4) loss edges and a (B_2, B_3) gain edge.

Identifying high probability CGRs

A breakpoint graph constructed from WGSS data will contain many alternating paths connecting candidate fused genes, and many alternating cycles. Some of the ambiguity arises because the WGSS data is produced from a diploid, or potentially poly-ploid, genome. Tumour chromosomes reassembled from copies of the same reference chromosomes will each produce a set of breakpoints. The WGSS data for these tumour chromosomes will then yield a merged set of all breakpoints. Only in very simplistic instances will it be possible to repartition the merged breakpoints into sets of breakpoints each produced by the same tumour chromosome. Furthermore, the breakpoints obtained from WGSS data will include a significant number of spurious predictions, especially when prioritizing sensitivity as proposed by nFuse. Spurious breakpoint predictions will further increase the number of alternating paths and cycles.

nFuse uses an objective function to identify real CGRs from the background of incidental and false positive structures. Our objective function is probabilistically motivated, and incorporates the probability that each breakpoint exists (breakpoint probability), in addition to a probability calculated for the total length of adjacency edges in the structure (CGR

length probability). Inclusion of the breakpoint probability will allow nFuse to mitigate the effects of spurious breakpoint predictions. We model the CGR length probability as an exponential distribution with scale parameter β, and motivate the choice of exponential independently for complex breakpoints/poly-fusions and CCBRs in the following sections. The negative log probability of a CGR with breakpoints X and adjacency edge lengths Y can be calculated as given in Equation 1, herein referred to as the CGR score.

$$\text{CGR score} \equiv -\log P(X,Y) \quad = \quad \log\beta + \sum_{y\in Y}\frac{y}{\beta} - \sum_{x\in X}\log P(x) \tag{1}$$

Let G be a breakpoint graph with breakpoint edges given distance -log $P(x)$, and adjacency edges given distance $\frac{y}{\beta}$. By inspection of Equation 1, an alternating cycle or path that maximizes $P(X,Y)$ will be a shortest alternating cycle or path on G.

Breakpoint prediction and probability estimation

We predict breakpoints from discordant paired end alignments. Our approach aims for high sensitivity by including reads with multiple genomic mappings, and reads that map only partially to the genome. To ensure adequate specificity, we calculate a probability for each breakpoint based on the alignment evidence and use that probability in downstream analysis including CGR discovery.

Let $\mathbf{R}$ be the set of paired end WGSS reads. We generate a set of mapping locations $\mathbf{M}$ for $\mathbf{R}$ using the following well established strategy (Tuzun et al., 2005; Volik et al., 2003). For each paired end read $(r_j^1, r_j^2) \in \mathbf{R}$:

1. identify a single concordant mapping location if it exists.
2. if no concordant mapping location exists:
   (a) identify the n top scoring mapping locations for r_j^1
   (b) identify the n top scoring mapping locations for r_j^2

We identify the n top scoring mapping locations for r_j^1 (and r_j^2) as follows. Let s_j be the maximum alignment score attained by *partial* alignment of read j to the genome. Briefly, a partial alignment is an alignment of the first ℓ nucleotides of the read (supplementary methods). Let k be the number of mappings of read j that attain s_j. If $k > n$ assume the read is unmappable and filter it, otherwise retain the k mapping locations.

Let $\mathbf{m}_j \in \mathbf{M}$ be the mapping locations identified for read $(r_j^1, r_j^2) \in \mathbf{R}$. Define the following indicator variables:

$$\begin{aligned} c_j &\equiv \text{read } j \text{ is concordant} \\ d_j &\equiv \text{the true alignment was discovered and is in the set } \mathbf{m}_j \end{aligned}$$

We make the assumption that reads mapped concordantly by the aligner are in fact concordant (with probability 1). We filter the concordantly mapped reads to create the set of discordant reads $\mathbf{R}^d$ and set of discordant mappings $\mathbf{M}^d$. As a result, $P(c_j = 1, d_j = 1) = 0$

for the set of filtered reads. We estimate probabilities for the remaining two possibilities for the true alignment of each read:

$$\begin{array}{lcl} P(c_j = 1|\cdot) & \equiv & \text{concordant but missed by the aligner} \\ P(d_j = 1|c_j = 0, \cdot) & \equiv & \text{discordant but missed by the aligner} \end{array}$$

We estimate $P(c_j = 1|\cdot)$ using the maximum concordant alignment score cs_j. To calculate cs_j, we align both ends of read j to all mapping locations in the set $\mathbf{m}_j$, and set cs_j to the maximum alignment score identified by this process. We then calculate $P(c_j = 1|cs_j)$ (supplementary methods), and use it to approximate $P(c_j = 1|\cdot)$. We approximate $P(d_j = 1|c_j = 0, \cdot)$ as $P(d_j = 1|c_j = 0, as_j)$ where as_j is the alignment score for read j (supplementary methods).

Next, we cluster the discordant alignments $\mathbf{M}^d$ based on the likelihood that a set of alignments were generated by the same breakpoint (supplementary methods). Let the resulting clusters of alignments represent putative breakpoints. Let g_{ij} indicate that putative breakpoint i generated read j. Assume $g_{ij} = 0$ if read j is not in the cluster that supports breakpoint i. We estimate $P(g_{ij} = 1|\cdot)$ as $P(g_{ij} = 1|nm_j, d_j = 1)$, where nm_j is the number of alternate mapping locations of read j. Under the assumption that all mapping locations discovered by the aligner are equally likely, we calculate $P(g_{ij} = 1|nm_j, d_j = 1) = \frac{1}{nm_j}$.

Finally, let b_i indicate that breakpoint i is true, let $\mathbf{G}_i$ be the set of all g_{ij} for breakpoint i, and let n_i be the number of reads that were generated by breakpoint i, that is $n_i = \sum_{g_{ij} \in \mathbf{G}_i} g_{ij}$. We estimate $P(b_i|n_i)$ (supplementary methods) and use it to estimate $P(b_i|\cdot)$ as given by equation 11.

$$\begin{aligned} P(b_i|\cdot) = \sum_{\mathbf{G}_i} P(b_i|n_i) \prod_j & P(g_{ij} = 1|nm_j, d_j = 1) \\ & \times P(d_j = 1|as_j, c_j = 0) \\ & \times P(c_j = 0|cs_j) \end{aligned} \tag{2}$$

Identifying high probability complex breakpoints and poly-fusions

Complex breakpoints and poly-fusions may be frequently occuring events in a rearranged tumour genome. Without further information, the biological significance of these events will be difficult to quantify. We use fusion transcripts predicted from RNA-seq to guide our search for complex breakpoints and poly-fusions, using effect on the transcriptome as an indicator of potential biological significance. The fusion transcripts also serve as a scaffold for reconstruction of the complex breakpoints / poly-fusions.

Given a gene A - gene B fusion transcript predicted from RNA-seq, we would like to predict the set of breakpoints that produced the A-B fusion. The breakpoints will often occur in the introns of gene A and B. As a result, these breakpoints are often spliced out of the A-B fusion transcript. Let x_A and x_B be the genomic positions of the splice sites in gene A and B that are predicted as spliced together in the fusion transcript. We would like to predict the intron sequence between x_A and x_B on the tumour chromosome. We model the intron lengths of fusion transcripts using an exponential with rate parameter β_p. An alternating path p from x_A to x_B represents a potential intron for the A-B fusion transcript,

and the total length of the adjacency edges in p equals the length of the putative intron. Following from the analysis that lead to the CGR score (Equation 1), we reconstruct the most likely intron by searching for the shortest alternating path between x_A and x_B on the graph G with $\beta = \beta_p$. See the supplementary methods for details on setting β_p.

Identifying high probability CCBRs

Very little is currently known about CCBRs, making model selection difficult. We model the total length of *loss* and *gain* adjacency edges in a CCBR using an exponential distribution with rate parameter β_c. We selected the exponential because it is the maximum entropy distribution for a positive random variate with fixed mean. For the purposes of this study, we have used $\beta_c = 2000bp$. We expect that the future discovery of additional CCBRs will allow us to properly estimate β_c.

Similar to complex breakpoints / poly-fusions, we search for CCBRs that are associated with fusion transcript predictions. For each breakpoint b that is part of a complex breakpoint / poly-fusion, we search for a CCBR that includes b. Following from the analysis that lead to the CGR score (Equation 1), we reconstruct the most likely CCBR that includes b by searching G with $\beta = \beta_c$ for the shortest alternating cycle that includes b. Specifically, we first remove the breakpoint edge (b_1, b_2) for breakpoint b from G, then search for the shortest between b_1 and b_2.

Results

We have used nFuse to identify CGRs in three datasets: a HCC1954 breast cancer cell line dataset, a dataset derived from primary tumour 963 (Wu et al., 2012), and a simulated dataset that includes 120 synthetic CGRs (see Table 1 for sequencing statistics). We used the HCC1954 dataset to assess breakpoint prediction sensitivity and breakpoint scoring specificity, and used the simulated dataset to assess precision and recall for CGR discovery. CGRs were retained only if their CGR score (Equation 1) was less than 20.

Table 1: Sequencing statistics for HCC1954 and 963.

	HCC1954		963		Simulation	
	WGSS	RNA-Seq	WGSS	RNA-Seq	WGSS	RNA-Seq
Read Length	36,80	36,50	50,76	50	80	50
Fragment Length Mean	193	176	406	233	300	250
Fragment Length Std. Dev.	37	33	49	36	50	40
Total Reads	340,977,703	175,508,350	176,764,897	86,720,870	208,839,566	2,877,519
Concordantly Mapped Reads	308,724,222	145,180,689	143,853,385	65,826,510	186,178,029	2,469,024

HCC1954 breast cancer cell line

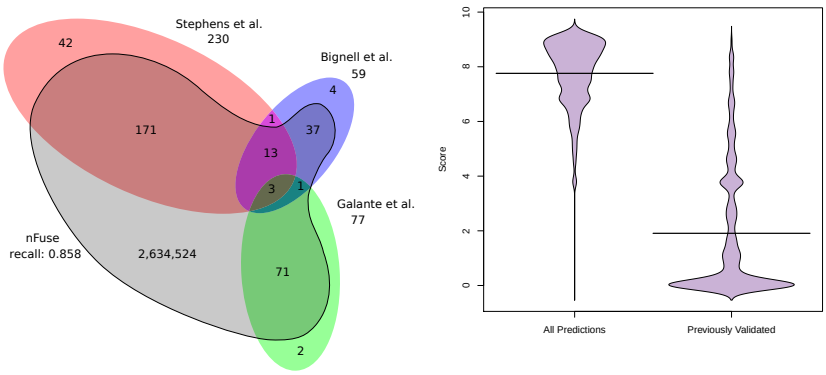
We first applied our method to publicly available data for HCC1954, a cell line that has been well studied at the molecular level. The HCC1954 cell line was derived from a ductal breast carcinoma and is estrogen receptor negative, progesterone receptor negative, and *ERBB2* positive (Zhao et al., 2010). Four recent studies sought to identify rearrangements

in HCC1954. Bignell et al. (2007) used end sequencing of bacterial artificial chromosome (BAC) libraries to discover rearrangements in tumour amplicons, and identified 59 unique breakpoints in HCC1954. Zhao et al. (2009), used long transcriptome reads and nominate fusion transcripts, and a combination of Long-Range PCR (LR-PCR) and fluorescence *in situ* hybridization (FISH) to identify underlying genomic rearrangements. Stephens et al. (2009) used WGSS to discover rearrangements in 24 breast cancers, and were able to identify 230 unique breakpoints in HCC1954. Some of the breakpoints discovered by Stephens et al. (2009) were more complex than a breakage and rejoining of two genomic loci. Interposed between the breakpoints were one or more *genomic shards*: small (<500bp) fragments of DNA from elsewhere in the genome. Galante et al. (2011), also used WGSS to discover somatic alterations in HCC1954, and identified 77 unique breakpoints. Finally, Asmann et al. (2011) used their pipeline SnowShoes-FTD to identify 4 fusion transcripts in HCC1954.

We obtained WGSS and RNA-seq data for HCC1954 from the NCBI Sequence Read Archive (SRA, `http://www.ncbi.nlm.nih.gov/Traces/sra`). The WGSS data (accession number ERA010917) is the same data used in the study by Galante et al. (2011), and the RNA-seq data (accession number ERA015355) was produced in a separate study on allele specific expression (Zhao et al., 2010). Next we compiled a list of 345 validated breakpoints from the studies by Bignell et al. (2007), Stephens et al. (2009), and Galante et al. (2011). Small deletions were excluded from the analysis since they are not the focus of this study. There was very little overlap between the sets of breakpoints discovered in each study, with only 3 breakpoints common to all three studies.

Using the previously validated breakpoints, we sought to estimate whether nFuse is sensitive enough to detect a significant proportion of real breakpoints, and whether the nFuse breakpoint ranking method could discern between real breakpoints and the background noise of spurious predictions. The breakpoint detection step of the nFuse pipeline identifies 296 of the 345 previously validated *true positive* breakpoints, accounting for 91.5% of the Bignell et al. (2007) breakpoints, 81.3% of the Stephens et al. (2009) breakpoints, and 97.4% of the Galante et al. (2011) breakpoints, for a recall of 0.858 (Figure 3a). In addition to the 296 true positive breakpoints, nFuse also identifies 2,634,524 additional breakpoints. Since 2,634,524 is well beyond the expected number of breakpoints in a rearranged tumour genome, and since we are including even very low probability breakpoint predictions, a large majority of the 2,634,524 are expected to be false positives. We sought to estimate whether the breakpoint probability we calculate could discriminate between true and false breakpoints. We first selected 3000 breakpoint predictions at random, and assumed that a significant majority of these predictions were false. Next we compared the scores of the 3000 randomly selected predictions to the scores of the true positive predictions (Figure 3b), finding that the true positive predictions scored significantly better than the randomly selected predictions (p-value $< 2.2 \times 10^{-16}$ Wilcoxon rank sum test).

Next we used a breakpoint graph constructed from the 2,634,524 HCC1954 breakpoints to predict CGRs in HCC1954 (see Table 2 and Table 3 for summaries of CCBRs and complex breakpoints respectively). We then attempted to validate the top 6 complex breakpoint / poly-fusion predictions as ranked by CGR score (Equation 1). For validation we performed Long Range PCR (LR-PCR) across the entire length of the complex breakpoint / poly-fusion. An event was considered validated if the size of the PCR product matched the predicted length, and Sanger sequencing of both ends of the PCR product



(a) (b)

Figure 3: Performance of nFuse breakpoint prediction on breakpoints previously discovered in HCC1954 (a) Shown is the overlap between sets of breakpoints discovered by Bignell et al. (2007), Stephens et al. (2009), Galante et al. (2011), and nFuse. Previously discovered breakpoints are rediscovered by nFuse with a recall of 0.858. (b) Beanplot comparing nFuse breakpoint scores for a random selection of 3000 nFuse breakpoint predictions, and the 296 'true positive' nFuse breakpoint predictions. Score is calculated as -log probability. The nFuse breakpoint scoring ranks true positive breakpoints significantly higher (closer to 0) than random breakpoints, many of which are expected to be false positives.

matched the predicted sequence. Five out of 6 LR-PCR assays produced PCR products, 4 of the predicted size. The PCR product for *PHF20L1-SAMD12* was approximately 1.5 kbp longer than expected. We confirmed by PCR that each of the 3 individual breakpoints predicted to form the *PHF20L1-SAMD12* poly-fusion were present in the *PHF20L1-SAMD12* PCR product. Thus we conclude that the *PHF20L1-SAMD12* prediction is correct but potentially incomplete, and suspect the existence of an additional insertion that is not identified when searching for the least complex solution. The 5 validated events are shown in Figure 4.

Table 2: Summary of putative CCBRs discovered in HCC1954. The CCBRs are grouped by number of breakpoints, and cumulative distance between breakpoints. Shown are counts for each group.

	Total Distance Between Breakpoints					
No. Breaks.	**0-500**	**500-1000**	**1000-2000**	**2000-5000**	**5000-10000**	**>10000**
2	6	0	3	5	2	1
3	6	1	4	1	1	0
4	1	2	0	0	3	3
5	0	0	0	0	0	4
6	0	0	0	0	0	1
7	0	0	0	0	0	2
8	0	0	0	0	1	0

Table 3: Summary of putative complex breakpoints discovered in HCC1954. The complex breakpoints are grouped by number of genomic shards, and total length of shards. Shown are counts for each group.

	Total Length of Genomic Shards					
No. Shards	0-500	500-1000	1000-2000	2000-5000	5000-10000	>10000
1	12	0	3	3	2	7
2	1	0	0	1	0	1
3	0	0	0	0	0	0
4	0	0	0	0	0	1
5	0	0	0	0	0	1
6	0	0	0	0	0	1

Four of the 5 validated complex breakpoints (Figures 4a-4d) express intergenic or intronic sequence, and are more likely to be truncating mutations than viable fusion genes. The fifth fusion transcript, *PHF20L1-SAMD12*, first discovered by Asmann et al. (2011), is predicted to preserve the reading frames of both *PHF20L1* and *SAMD12* (Figure 4e). Among high confidence fusion transcripts, *PHF20L1-SAMD12* expression is second only to *STRADB-NOP58*, as suggested by read depth at the breakpoint.

Genomic shards have implications for traditional rearrangement detection techniques. The genomic shards range in size from 220bp to 4.3 kbp for the 5 events we validated. A 220 bp genomic shard may be small enough such that some paired end reads span the full complex breakpoint, allowing detection of the breakpoint using conventional methods. However, the 1 kbp and larger genomic shards will be longer than the paired end reads in most WGSS assays, preventing straightforward detection of the breakpoint. Thus a potentially interesting gene fusion such as *PHF20L1-SAMD12* would be impossible to discover when considering only single breakpoints as evidence for gene fusions, as has been done previously (Bashir et al., 2008). Instead, such methods would falsely nominate *SAMD12-FAM49B* and *PHF20L1*-Intergenic as truncating mutations.

We identified an additional 2 high-confidence poly-fusions by searching for sequences of genomic shards *highly connected* by fusion transcripts. For each fusion transcript corroborated by alternating path p, we searched for other fusion transcripts corroborated by a subpath of p. We used the resulting sets of (non-conflicting) fusion transcripts to identify poly-fusions for which each genomic shard is expressed in at least one non-conflicting fusion transcript. Thus we use fusion transcripts as a scaffold for local genome reconstruction. We believe the highly connected nature of the additional 2 poly-fusions (Figure 5) provides more confidence in these events.

Finally, we used the 7 complex breakpoints / poly-fusioins (5 validated and 2 high-confidence) to evaluate the utility of including suboptimal breakpoint predictions in the breakpoint graph. Statistics for the 7 events are detailed in Table 4. The 15 breakpoints for the 7 events include breakpoints with low read support, breakpoints supported by multi-map reads, and low probability breakpoints. Three of the breakpoints are supported by only 1 read. For 2 of the breakpoints, the entire set of supporting reads also align to other genomic loci, and form a coherent cluster at those loci. Thus even multi-map resolution methods such as VariationHunter (Hormozdiari et al., 2009) may be unable to identify the correct mappings of these reads. Another 2 breakpoints are given low probability due to

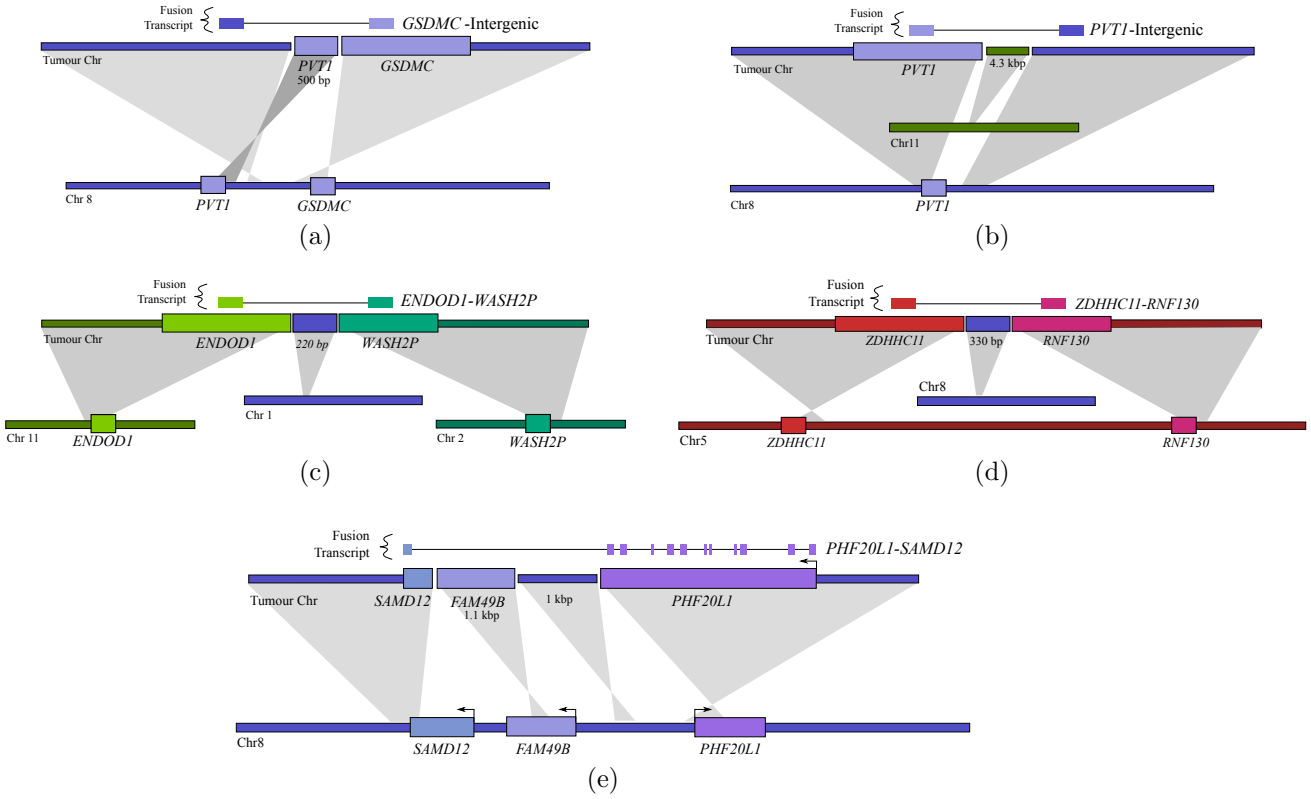


Figure 4: Complex breakpoints and poly-fusions predicted by nFuse and validated by LR-PCR. (a-d) Complex breakpoints produce truncated *GSDMC* and *PVT1* transcripts, and *ENDOD1-WASH2P* and *ZDHHC11-RNF130* fusion transcripts. (e) A *PHF20L1-FAM49B-SAMD12* poly-fusion produces an in-frame *PHF20L1-SAMD12* fusion transcript.

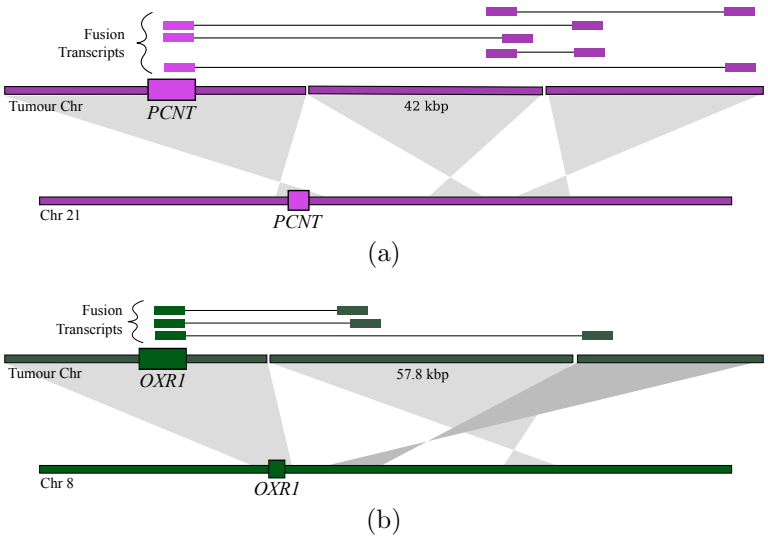


Figure 5: Complex breakpoints and poly-fusions corroborated by multiple fusion transcripts

the existence of marginal concordant alignments. Using breakpoints supported by at least 2 uniquely aligning, high confidence discordant reads would have resulted in identification of only 3 of the 7 events.

Table 4: Statistics for CGR breakpoints detected by nFuse. Columns: *Type*: Closed Chain Breakage and Rejoining (CCBR) or Complex Breakpoint (CB) *CGR score*: score calculated as per Equation 1, *breakpoint*: order of the breakpoint in the CCBR or CB, *read count*: number of supporting WGSS reads, *multi-map*: average number of genomic loci to which supporting reads can alternatively be mapped, *probability*: breakpoint probability as calculated using Equation 11, *score*: negative log of the breakpoint probability, *rank*: rank of the breakpoint in the set of all breakpoints ordered by score.

CGR	Case	Type	CGR score	breakpoint	read count	multi-map	probability	score	rank
HMGN2P46-MYC	963	CCBR	12.06	1	5	1.8	0.994	0.01	613
				2	3	1.3	0.022	3.83	4,328
				3	6	1.0	1.000	0.00	414
				4	1	1.0	0.001	7.47	152,379
WDTC1-EFCAB4A	963	CCBR/CB	7.42	1	6	1.0	0.759	0.28	849
				2	1	1.0	0.001	6.59	48,415
				3	12	1.3	1.000	0.00	541
ZDHHC11-RNF130	*HCC1954*	*CB*	1.74	1	10	1.0	0.545	0.61	1,311
				2	9	1.7	0.382	0.96	1,574
ENDOD1-WASH2P	*HCC1954*	*CB*	4.75	1	5	1.8	0.998	0.00	400
				2	14	5.1	0.578	0.55	1,277
PVT1-Intergenic	*HCC1954*	*CB*	2.96	1	3	1.3	0.465	0.76	1,414
				2	5	1.0	0.630	0.46	1,204
PHF20L1-SAMD12	*HCC1954*	*CB*	7.31	1	1	1.0	0.023	3.78	16,445
				2	9	3.0	1.000	0.00	273
				3	3	1.0	0.074	2.60	4,102
GSDMC-Intergenic	*HCC1954*	*CB*	9.04	1	1	1.0	0.004	5.58	139,931
				2	3	1.0	0.059	2.83	4,576
OXR1-Intergenic	*HCC1954*	*CB*	14.29	1	11	2.5	1.000	0.00	204
				2	2	1.5	0.026	3.65	6,879
PCNT-Intergenic	*HCC1954*	*CB*	16.77	1	1	1.0	0.001	6.63	400,357
				2	2	2.5	0.081	2.52	3,908

Simulated Dataset

We used a simulated dataset to estimate the sensitivity of the nFuse method. We generated 209 million 80 × 80 WGSS reads and 2.9 million 50 × 50 RNA-seq reads from a simulated genome that included 60 CCBRs and 60 complex breakpoints. WGSS and RNA-seq reads were generated using `maq simulate`, and simulation parameters trained from the HCC1954 data (lanes ERR016395 and ERR022661). The 60 CCBRs and 60 complex breakpoints were generated in 3 different classes of difficulty, with features of each CGR selected uniformly and at random from a range of values dependent on the class. We analyzed the simulated dataset using nFuse with a threshold of 20 for the CGR score.

For the complex breakpoints, we first selected 2 genes with at least one intron each, then selected an intron from each gene. We created a fusion transcript by splicing the 5' exons of the first gene to the 3' exons of the second gene, and sampled RNA-seq reads from the fusion transcript at a coverage selected from between 20× and 200×. Next we created a

complex breakpoint composed of n number of shards of length $\{\ell_1..\ell_n\}$, where n and $\{\ell_1..\ell_n\}$ were selected from a range of values dependent on the class difficulty (Table 5). WGSS reads were sampled from the complex breakpoint at a coverage selected from between 5× and 30×. nFuse detected 49 of 60 complex breakpoints in the simulated dataset (Table 5).

Table 5: CCBRs identified in a simulated dataset.

Class	Shard Length	Num. Shards	Recall
A	500-2000	1	19/20
B	1000-2000	2-4	16/20
C	2000-10000	3-5	19/20
		Total	54/60

For the CCBRs, we again selected 2 genes, created a fusion transcript, and generated reads as described above for complex breakpoints. We then simulated a simple breakpoint between the two genes. We also simulated an additional $n-1$ breakpoints with the structure of a CCBR. Each breakpoint was separated by a distance ℓ_i from the subsequent breakpoint in the CCBR. Values for n and $\{\ell_1..\ell_n\}$ were selected from a range of possibilities that depended on the class of difficulty (Table 6). WGSS reads were sampled from the n breakpoints at a coverages selected from between 5× and 30×. nFuse detected 54 of 60 complex breakpoints in the simulated dataset (Table 6).

Table 6: Complex breakpoints identified in a simulated dataset.

Class	Max. Dist.	Num. Breaks.	Recall
A	500	2-3	17/20
B	1000	4-5	18/20
C	2000	6-7	14/20
		Total	49/60

nFuse predicts an additional 3 CCBRs and 4 complex breakpoints in the simulated dataset. For 3 of the 4 false positive complex breakpoints, the predicted sequence is identical to the sequence of an undiscovered simulated complex breakpoint. However, for each of these 3 predictions, at least one of the shards is predicted to originate from the wrong location in the genome. Instead, a homologous region is incorrectly predicted as the origin of those shards. The remaining complex breakpoint and 3 CCBRs also represent undiscovered simulated events with misplaced breakpoints due to homology. Based on the simulation, we estimate the precision of nFuse to be 0.92 for complex breakpoints and 0.95 for CCBRs.

Primary prostate tumour 963

We applied nFuse to the discovery of complex rearrangements in sample 963, a primary prostate tumour sample. We generated WGSS data at 17× physical coverage in addition to 150 million reads of matched RNA-seq (Wu et al., 2012). The WGSS data produced 762,675 putative breakpoint predictions, and these were used to construct the breakpoint graph for 963. The primary CGR feature of 963 was CCBRs (see Table 7 and Table 8 for summaries of CCBRs and complex breakpoints respectively). A 4-loci CCBR and 3-loci

CCBR CCBRs were prioritized for validation since both affected cancer relevant genes, and each contained a breakpoint supported by only a single read. We validated all breakpoints and fusion transcripts associated with both CCBRs. We also validated a complex breakpoint associated with the 3-loci CCBR using LR-PCR. The complex breakpoint and two CCBRs are described in detail below.

Table 7: Summary of putative CCBRs discovered in 963. The CCBRs are grouped by number of breakpoints, and cumulative distance between breakpoints. Shown are counts for each group.

	Total Distance Between Breakpoints					
No. Breaks	**0-500**	**500-1000**	**1000-2000**	**2000-5000**	**5000-10000**	**>10000**
2	7	5	2	5	4	0
3	1	1	1	2	0	0
4	1	1	1	0	1	0
5	0	0	0	0	0	1

Table 8: Summary of putative complex breakpoints discovered in 963. The complex breakpoints are grouped by number of genomic shards, and total length of shards. Shown are counts for each group.

	Total Length of Genomic Shards					
No. Shards	**0-500**	**500-1000**	**1000-2000**	**2000-5000**	**5000-10000**	**>10000**
1	4	1	0	0	0	2
2	1	0	0	0	0	1

As described by Wu et al. (2012), the 963 tumour is significant because it is difficult to classify in the context of established prostate cancer biology. Although the histology of 963 is consistent with a uniform cell type, the gene expression profile is suggestive of a hybrid luminal/neuroendocrine phenotype. The fusion genes discovered in 963 exhibit a similar hybrid pattern. Some of the fused genes are primarily expressed in luminal cells, and others are primarily expressed in neuroendocrine cells. A growing body of evidence suggests that the binding of transcriptional machinery predisposes DNA to double stranded breaks (Lin et al., 2009; Nambiar and Raghavan, 2011). Thus Wu et al. (2012) hypothesized that luminal and neuroendocrine expression patterns were present in nascent 963 tumour cells.

Among the catalogue of luminal/neuroendocrine fusion genes discovered in 963, two are of particular interest because of their association with a CGR. Most notable of these fusions is highly expressed *HMGN2P46-MYC*, a promoter exchange between the *MYC* oncogene and luminal cell specific *HMGN2P46*, with a breakpoint in *MYC* similar to that found in Burkitt's Lymphoma (Dave et al., 2006). Seemingly unrelated is the *ARHGEF17-SHANK2* fusion involving neuroendocrine specific *SHANK2*, previously reported as fused in melanoma (Berger et al., 2010). We have discovered and validated a CCBR consisting of 4 breakpoints, one of which produces a *HMGN2P46-MYC* fusion and another that produces a *ARHGEF17-SHANK2* fusion (Figure 6a). The discovery of a single genomic event that produces two fusion transcripts, one involving a luminal specific *HMGN2P46*, and another involving neuroendocrine specific *SHANK2*, is evidence that tumourigenesis occurred in a progenitor cell simultaneously expressing both luminal and neuroendocrine specific genes.

We have also identified a CGR that is represented in the breakpoint graph by a path and a cycle. The path represents a complex breakpoint involving the *WDTC1*, *PRKRIP1*, and *EFCAB4A* genes on chromosomes 1, 7, and 11 respectively. The complex breakpoint was identified as the underlying genomic rearrangement explaining several fusion transcripts. nFuse also identifies a cycle that uses the same two breakpoints as the path, and one additional breakpoint. Given all available information, including fusion transcripts and the 3 breakpoints, the most parsimonious CGR is a reciprocal translocation between chromosomes 1 and 11, with an insertion of a 800bp shard of chromosome 7 at one of the breakpoints (Figure 6b). Without knowledge of the fusion transcripts, and given previous interpretation of CCBRs (Berger et al., 2011), the breakpoints may have been interpreted differently. Specifically, the 3 breakpoints could alternatively represent a transformation that produces 3 tumour chromosomes, a 1-7 chromosome, a 7-11 chromosome, and an 11-1 chromosome. We are able to exclude this alternate possibility by using the fusion transcripts as a scaffold for local reconstruction of the CGR.

As mentioned in reference to Figure 2b, *gain* adjacency edges produce CCBRs with ambiguous structures. The gain adjacency edge for *WDTC1-PRKRIP1-EFCAB4A* was determined to represent an insertion of a shard of chromosome 7 at a breakpoint between chromosomes 1 and 11, rather than a region of chromosome 7 duplicated in two tumour chromosomes. Thus we sought to rule out the possibility that *all* gain adjacency edges represent insertions. The *MYC* CCBR includes two gain adjacency edges. If either of these gain edges represented insertions, it would be impossible for the *MYC* CCBR alone to explain the *ARHGEF17-SHANK2* and *HMGN2P46-MYC* fusion transcripts; additional breakpoints and further complexity would be required. Thus the most parsimonious explanation is that the gain adjacency edges of the *MYC* CCBR represent regions of chromosome 11 that are duplicated in the final tumour chromosomes.

Discussion

We have applied nFuse to the discovery of fusion transcripts and underlying Complex Genomic Rearrangements (CGRs) in breast cancer cell line HCC1954 and a primary prostate tumour sample 963. The landscape of CGR events differed between HCC1954 and 963, with complex breakpoints and poly-fusions arising as the predominant CGR feature of HCC1954, and closed chains of breakage and rejoining (CCBRs) arrising as the predominant feature of 963. In HCC1954, nFuse predicted 7 high confidence complex breakpoints / poly-fusions. One of these fusions is *PHF20L1-SAMD12*, a highly expressed in-frame fusion missed by Stephens et al. (2009) likely because of the complexity of the breakpoint. In fact, our results strongly suggest that analysis of single breakpoints in isolation is inadequate as a method for identifying fusion genes. The large size of fragments that are interposed at the breakpoints of some fusion genes will prevent the discovery of those fusions. nFuse is also capable of identifying the single breakpoints underlying fusion transcripts caused by more simple rearrangements. nFuse successfully recovers all 4 fusion transcripts identified by ShowShoes-FTD, predicting a simple rearrangement for three and a CGR for the forth.

In 963, nFuse identified a CGR with potential biological implications. Based on existing evidence that transcribed genes are prone to double stranded breaks, and building on the

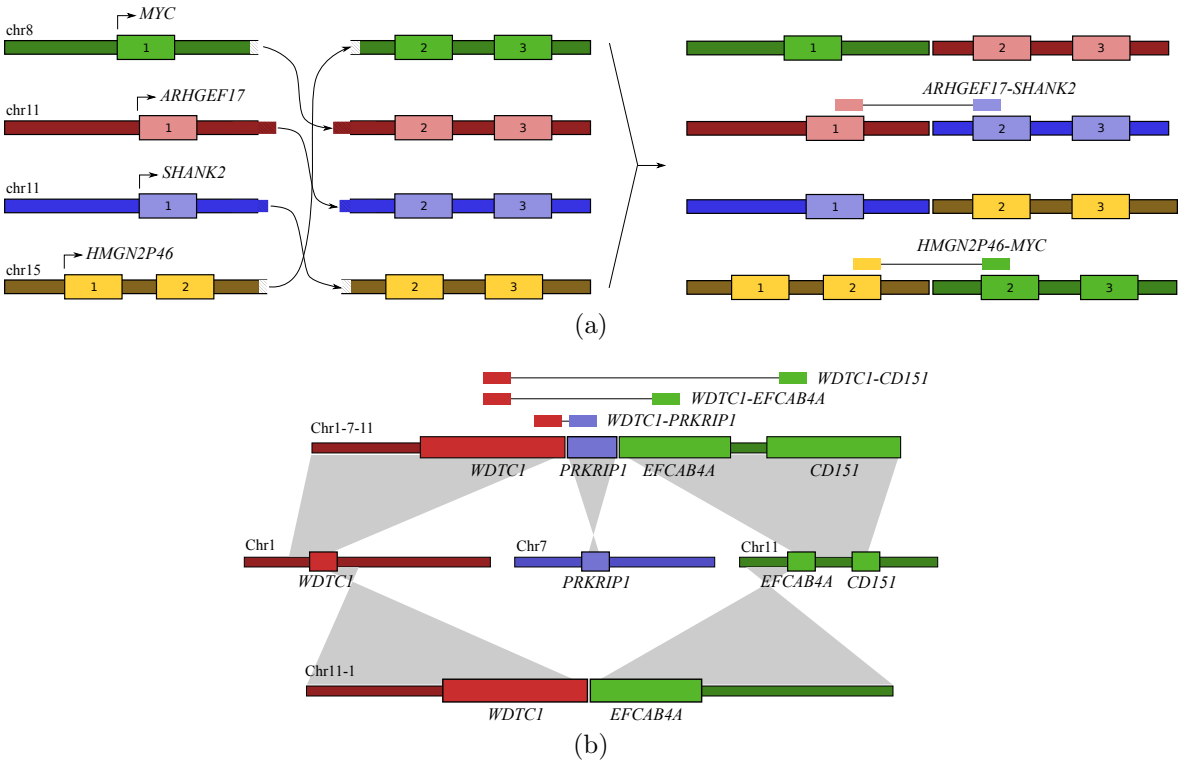


Figure 6: CGRs discovered in primary tumour sample 963. (a) A single CCBR produces 4 fusion genes: *MYC-ARHGEF17*, *ARHGEF17-SHANK2*, *SHANK2-HMGN2P46*, and *HMGN2P46-MYC*. Only the *ARHGEF17-SHANK2* and *HMGN2P46-MYC* fusion genes produce fusion transcripts. (b) Example of a CGR that is both a CCBR and a fusion involving 3 loci. The aberrant 1-7-11 chromosome produces 3 fusion transcripts: *WDTC1-CD151*, *WDTC1-EFCAB4A*, and *WDTC1-PRKRIP1*.

suggestion by Berger et al. (2011) that CCBRs occur for sets of genes recruited to the same transcriptional factory, we propose that CCBRs may be used to infer the gene co-expression history of a tumour. In 963, the discovery of the *MYC* CCBR suggests that luminal specific *HMGN2P46* and neuroendocrine specific *SHANK2* were co-expressed in a single nascent tumour cell during the formation of the CCBR. Thus the CCBR provides further evidence of the dual luminal/neuroendocrine history of the 963 tumour, and suggests the unusual luminal/neuroendocrine expression pattern of the tumour predates the formation of the *MYC* rearrangement.

We have used examples in both HCC1954 and 963 to highlight the potential utility of performing an integrated analysis of matched WGSS and RNA-seq datasets. The RNA-seq data yields information about long range connectivity between genomic regions, acting as a set of very long genomic reads. As such, the RNA-seq data can be useful as a scaffold for reconstructing tumour chromosomes. In some cases the RNA-seq data can also be used to resolve genomic architectural ambiguities, as for the 2 CCBRs discussed for 963. Finally, RNA-seq can be used to identify potentially interesting events such as fusion transcripts that serve as a starting point for targeted analysis.

Many fusion genes, including those with complex origins such as *PHF20L1-SAMD12*, can be detected using conventional analysis of RNA-seq data. Nevertheless, many interesting questions cannot be answered with a fusion transcript prediction alone. For instance, it is impossible to measure the clonal abundance of a gene fusion at the transcriptomic level alone, since transcript abundance is heavily influenced by expression. Given multiple tumour samples from the same patient, knowledge of breakpoints will allow us to ask which samples harbour the fusion, whereas knowledge of the fusion transcript only allows us to understand the expression levels in each sample. Furthermore, an understanding of the clonal abundances of rearrangements will help determine the evolutionary history of the tumour. The evolutionary history will then help determine the founder status of each rearrangement, and which rearrangements are drivers of tumourogenesis (Shah et al., 2012).

Finally, it has been assumed throughout this work that the breakpoints of CGRs occur simultaneously during a single event. A complex breakpoint with one genomic shard could also be formed by a two independent breakage-rejoining events that occur at the same loci at different times during the tumours development. Similarly, a CCBR could be formed by breakage and rejoining events occuring in succession at the same loci. Both of these scenarios require the formation of intermediate breakpoints. In the evolutionary history of the tumour, some cells would likely have evolved from cells in the intermediate state without having gained all breakpoints in the CGR. Thus we expect the intermediate breakpoints to be present in some proportion of tumour cells, though that proportion may be very small. To date we have not identified any intermediate breakpoints in the sequencing data. Future work will involve testing for intermediate breakpoints, and negative results will provide further evidence of the simultaneity of CGRs.

Data Access

The nFuse source code and manual can be downloaded at
`http://n-fuse.sourceforge.net/`

Acknowledgments

References

Asmann, Y. W., Hossain, A., Necela, B. M., Middha, S., Kalari, K. R., Sun, Z., Chai, H. S., Williamson, D. W., Radisky, D., Schroth, G. P., *et al.*, 2011. A novel bioinformatics pipeline for identification and characterization of fusion transcripts in breast cancer and normal cell lines. *Nucleic Acids Res*, **39**(15).

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

Berger, M. F., Lawrence, M. S., Demichelis, F., Drier, Y., Cibulskis, K., Sivachenko, A. Y., Sboner, A., Esgueva, R., Pflueger, D., Sougnez, C., *et al.*, 2011. The genomic complexity of primary human prostate cancer. *Nature*, **470**(7333):214–220.

Berger, M. F., Levin, J. Z., Vijayendran, K., Sivachenko, A., Adiconis, X., Maguire, J., Johnson, L. A., Robinson, J., Verhaak, R. G., Sougnez, C., *et al.*, 2010. Integrative analysis of the melanoma transcriptome. *Genome Res*, **20**(4):413–427.

Bignell, G. R., Santarius, T., Pole, J. C., Butler, A. P., Perry, J., Pleasance, E., Greenman, C., Menzies, A., Taylor, S., Edkins, S., *et al.*, 2007. Architectures of somatic genomic rearrangement in human cancer amplicons at sequence-level resolution. *Genome Res*, **17**(9):1296–1303.

Brown, J. R., 1974. Shortest alternating path algorithms. *Networks*, **4**:311–334.

Chen, K., Wallis, J. W., McLellan, M. D., Larson, D. E., Kalicki, J. M., Pohl, C. S., McGrath, S. D., Wendl, M. C., Zhang, Q., Locke, D. P., *et al.*, 2009. BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nat Methods*, **6**(9):677–681.

Dave, S. S., Fu, K., Wright, G. W., Lam, L. T., Kluin, P., Boerma, E. J., Greiner, T. C., Weisenburger, D. D., Rosenwald, A., Ott, G., *et al.*, 2006. Molecular diagnosis of burkitt's lymphoma. *N Engl J Med*, **354**(23):2431–2442.

Elkan, C. and Noto, K., 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 213–220, New York, NY, USA. ACM.

Galante, P., Parmigiani, R., Zhao, Q., Caballero, O., de Souza, J., Navarro, F., Gerber, A., Nicolás, M., Salim, A., Silva, A., *et al.*, 2011. Distinct patterns of somatic alterations in a lymphoblastoid and a tumor genome derived from the same individual. *Nucleic Acids Research*, **39**(14):6056–6068.

Gotoh, O., 1982. An improved algorithm for matching biological sequences. *Journal of molecular biology*, **162**(3):705–708.

Greenman, C. D., Pleasance, E. D., Newman, S., Yang, F., Fu, B., Nik-Zainal, S., Jones, D., Lau, K. W., Carter, N., Edwards, P. A., *et al.*, 2011. Estimation of rearrangement phylogeny for cancer genomes. *Genome Res*, .

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

Langmead, B. and Salzberg, S., 2012. Fast gapped-read alignment with bowtie 2. *Nat Meth*, **9**(4):357–359.

Lee, C.-H., Ou, W.-B., Mariño-Enriquez, A., Zhu, M., Mayeda, M., Wang, Y., Guo, X., Brunner, A., Amant, F., French, C., *et al.*, 2012. 14-3-3 fusion oncogenes in high-grade endometrial stromal sarcoma. *Proceedings of the National Academy of Sciences of the United States of America*, **109**(3):929–934.

Lin, C., Yang, L., Tanasa, B., Hutt, K., Ju, B., Ohgi, K., Zhang, J., Rose, D., Fu, X., Glass, C., *et al.*, 2009. Nuclear receptor-induced chromosomal proximity and dna breaks underlie specific translocations in cancer. *Cell*, **139**(6):1069–1083.

McPherson, A., Hormozdiari, F., Zayed, A., Giuliany, R., Ha, G., Sun, M., Griffith, M., Heravi Moussavi, A., Senz, J., Melnyk, N., *et al.*, 2011a. defuse: An algorithm for gene fusion discovery in tumor rna-seq data. *PLoS Comput Biol*, **7**(5):e1001138.

McPherson, A., Wu, C., Hajirasouliha, I., Hormozdiari, F., Hach, F., Lapuk, A., Volik, S., Shah, S., Collins, C., and Sahinalp, C., *et al.*, 2011b. Comrad: detection of expressed rearrangements by integrated analysis of rna-seq and low coverage genome sequence data. *Bioinformatics*, **27**(11):1481–1488.

Mitelman, F., Johansson, B., and Mertens, F., 2007. The impact of translocations and gene fusions on cancer causation. *Nat Rev Cancer*, **7**(4):233–245.

Nambiar, M. and Raghavan, S., 2011. How does dna break during chromosomal translocations? *Nucleic Acids Research*, **39**(14):5813–5825.

Ozery-Flato, M. and Shamir, R., 2009. Sorting cancer karyotypes by elementary operations. *J Comput Biol*, **16**(10):1445–1460.

Pevzner, P., 2000. *Computational Molecular Biology: An Algorithmic Approach (Computational Molecular Biology)*. The MIT Press.

Raphael, B. J. and Pevzner, P. A., 2004. Reconstructing tumor amplisomes. *Bioinformatics*, **20 Suppl 1**:265–273.

Raphael, B. J., Volik, S., Collins, C., and Pevzner, P. A., 2003. Reconstructing tumor genome architectures. *Bioinformatics*, **19 Suppl 2**:162–171.

Shah, S., Roth, A., Goya, R., Oloumi, A., Ha, G., Zhao, Y., Turashvili, G., Ding, J., Tse, K., Haffari, G., *et al.*, 2012. The clonal and mutational evolution spectrum of primary triple-negative breast cancers. *Nature*, **advance online publication**.

Stephens, P., Greenman, C., Fu, B., Yang, F., Bignell, G., Mudie, L., Pleasance, E., Lau, K., Beare, D., Stebbings, L., *et al.*, 2011. Massive genomic rearrangement acquired in a single catastrophic event during cancer development. *Cell*, **144**(1):27–40.

Stephens, P. J., McBride, D. J., Lin, M. L., Varela, I., Pleasance, E. D., Simpson, J. T., Stebbings, L. A., Leroy, C., Edkins, S., Mudie, L. J., *et al.*, 2009. Complex landscapes of somatic rearrangement in human breast cancer genomes. *Nature*, **462**(7276):1005–1010.

Tuzun, E., Sharp, A. J., Bailey, J. A., Kaul, R., Morrison, V. A., Pertz, L. M., Haugen, E., Hayden, H., Albertson, D., Pinkel, D., *et al.*, 2005. Fine-scale structural variation of the human genome. *Nat Genet*, **37**(7):727–732.

Volik, S., Zhao, S., Chin, K., Brebner, J. H., Herndon, D. R., Tao, Q., Kowbel, D., Huang, G., Lapuk, A., Kuo, W. L., *et al.*, 2003. End-sequence profiling: sequence-based analysis of aberrant genomes. *Proc Natl Acad Sci U S A*, **100**(13):7696–7701.

Wu, C., Wyatt, A. W., Lapuk, A. V., McPherson, A., McConeghy, B. J., Bell, R. H., Anderson, S., Haegert, A., Brahmbhatt, S., Shukin, R., *et al.*, 2012. Integrated genome and transcriptome sequencing identifies a novel form of hybrid and aggressive prostate cancer. *J Pathol*, **227**(1):53–61.

Zhao, Q., Caballero, O. L., Levy, S., Stevenson, B. J., Iseli, C., de Souza, S. J., Galante, P. A., Busam, D., Leversha, M. A., Chadalavada, K., *et al.*, 2009. Transcriptome-guided characterization of genomic rearrangements in a breast cancer cell line. *Proc Natl Acad Sci U S A*, **106**(6):1886–1891.

Zhao, Q., Kirkness, E., Caballero, O., Galante, P., Parmigiani, R., Edshall, L., Kuan, S., Ye, Z., Levy, S., Vasconcelos, A., *et al.*, 2010. Systematic detection of putative tumor suppressor genes through the combined use of exome and transcriptome sequencing. *Genome Biology*, **11**(11):R114.

Supplemental

nFuse pipeline overview

The nFuse method builds upon Comrad (McPherson et al., 2011b), our previous work on rearrangement detection in matched RNA-seq and WGSS. We begin this section by briefly describing Comrad, then describe significant differences between Comrad and nFuse. An overview of the nFuse pipeline is shown in Figure 7.

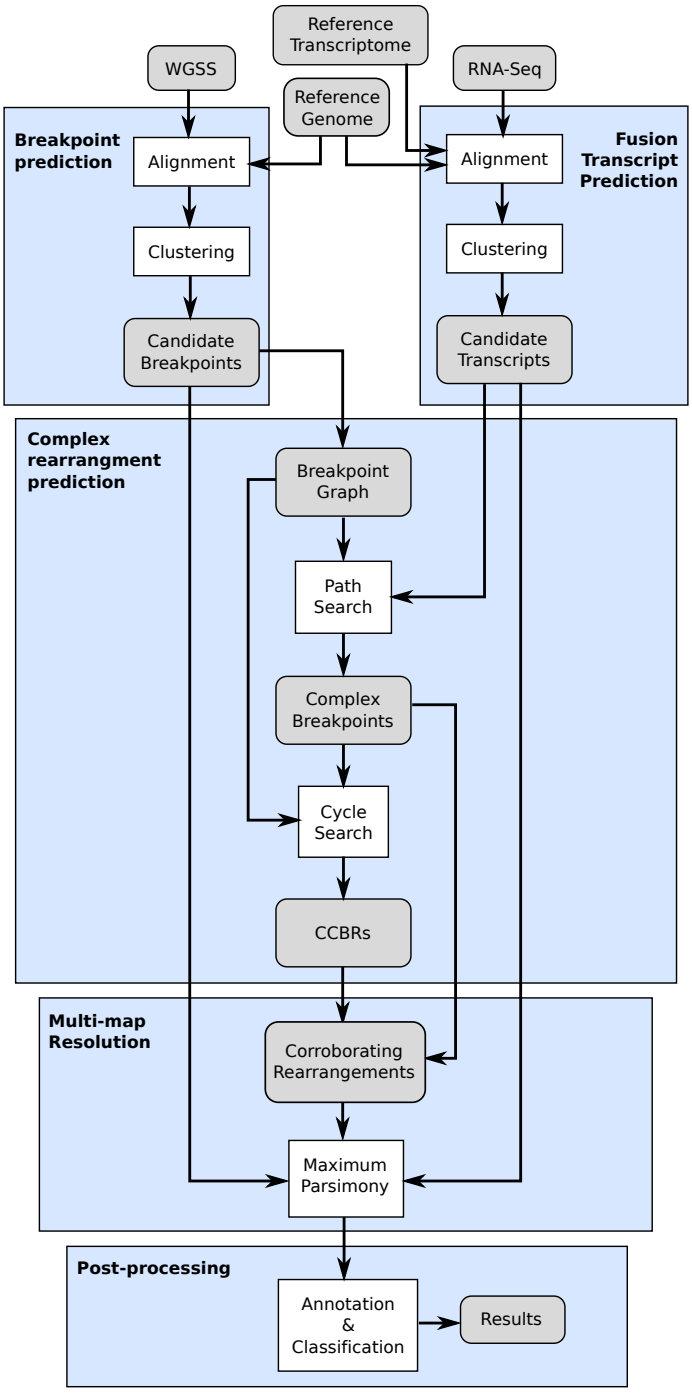


Figure 7: **Overview of the nFuse pipeline.** Generally, the nFuse pipeline involves 5 major steps: breakpoint prediction, fusion transcript prediction, complex rearrangement prediction, multi-map resolution, and post-processing. Shaded nodes represent data and unshaded nodes represent analysis.

Comrad was developed to predict fusion transcripts and their associated rearrangements from matched RNA-seq and WGSS tumour data. Comrad begins by aligning RNA-seq reads to the reference transcriptome and WGSS reads to the reference genome. Paired end RNA-

seq reads for which both ends align to the same gene are classified as concordant, and all other reads are classified as discordant. Discordant RNA-seq reads are clustered into sets of reads that suggest the same candidate fusion transcripts. Paired end WGSS reads for which both ends align in the expected direction within 2kb are classified as concordant, and all other reads are classified as discordant. Discordant WGSS reads are clustered into sets of reads that suggest the same candidate breakpoint.

The key contribution provided by Comrad was a method for resolving multiply mapped RNA-seq and WGSS reads using a maximum parsimony based combinatorial formulation. Real and biologically relevant gene fusions are known to exist where one of the fusion partners shares significant sequence homology with other genes in the genome (Lee et al., 2012). These fusions may produce RNA-seq reads with ambiguous genomic origin, suggesting multiple, equally likely fusion transcripts. If a corroborating breakpoint can be predicted from WGSS data, it may be possible to identify the actual fusion transcript. For example, if A-B and A-B' are two fusion transcripts suggested by the same set of multi-mapping RNA-seq reads, an A-B' breakpoint predicted from matched WGSS data would help to identify A-B' as more likely. The reverse is also true: unambiguous RNA-seq reads can assist in predicting the correct breakpoint implied by multi-mapping WGSS reads. In fact, even for ambiguous RNA-seq and WGSS reads, it should be possible to correctly associate the RNA-seq and WGSS evidence even though the exact gene pair remains unknown. Unfortunately, naive application of Comrad to the identification of CGRs will result in over-prediction of these events.

The nFuse pipeline differs from Comrad in 4 major areas: WGSS alignment, discordant read clustering, corroboration between fusion transcripts and breakpoints, and the maximum parsimony formulation. WGSS reads are aligned using a seed and extend strategy, and the best *partial* discordant alignments of the WGSS reads are used to predict breakpoints. Discordant reads are clustered using a mixture model and the EM algorithm. nFuse adds detection of CGRs associated with fusion transcripts, replacing the Comrad method of corroborating fusion transcripts and breakpoints. Finally, nFuse incorporates a new maximum parsimony formulation for resolving multi-map reads that does not over-predict CGRs.

Partial alignments of WGSS reads

As sequencing technology improves and read lengths increase, a larger proportion of each DNA fragment is sequenced, and a smaller proportion of the fragment remains unsequenced. Thus it becomes increasingly likely that a breakpoint will fall within a sequenced region of a DNA fragment rather than the unsequenced region in the middle of a fragment. As a result we will be less likely to find a complete and contiguous alignment of both reads produced by DNA fragment harbouring a breakpoint. For instance, in HCC1954, the DNA fragments are approximately 193 bp in length, and many of the reads are 81 bp in length. If we expect complete and contiguous alignments, we will only be able to identify discordant reads for which the breakpoint is in the $193 - 2 \times 81 = 31$bp region in the middle of the read.

To mitigate the aforementioned problem, we search for *partial* alignments of discordant paired end WGSS reads. Let r be the sequence of one end of a paired end read and define the partial alignment of r as an alignment of the first ℓ nucleotides of r where $1 \leq \ell \leq |r|$. A read with a breakpoint at position $\ell + 1$ in the read should ideally produce a partial alignment

of length ℓ. The score of a partial alignment is calculated using a fixed bonus for matches, an affine gap penalty, and penalty for mismatches based on the quality of the read at the mismatch. A partial alignment can be calculated using a trivial modification to the dynamic programming algorithm for calculating alignments with affine gap penalties (Gotoh, 1982).

We use bowtie2 in local alignment mode as an approximate but effective method for generating partial alignments of reads (Langmead and Salzberg, 2012). To generate the n top scoring mapping locations for a read, we first use bowtie2 with parameters `--very-sensitive-local -k` $n+1$ to calculate $n+1$ local alignments, and re-score these alignments if they include soft-clipping at the beginning of the read. We use the default scoring method implemented by bowtie2 for end-to-end alignments, re-described here. A gap of length N is given a penalty calculated as,

$$\text{GO} + N \times \text{GE}.$$

We use the bowtie2 defaults, GO $= 5$ and GE $= 3$. Mismatches are given a penalty calculated as,

$$\text{MN} + \text{floor}\left(\frac{\text{MX} - \text{MN}}{\frac{1}{40}\min(Q, 40.0)}\right)$$

where Q is the Phred quality value. We use the bowtie2 defaults, MN $= 2$ and MX $= 6$. Matches are given a bonus MA $= 2$, the default for bowtie2. Let $S = \{s_1, s_2, ..., s_n, s_{n+1}\}$ be the resulting set of scores and let $T = \{s_i : s_i = \max(S)\}$ be the set of scores that attain the maximum value. If $|T| = n + 1$ the read is filtered, otherwise the set of alignments T is retained. By default nFuse uses $n = 20$.

Discordant read clustering

Discordant reads are clustered into sets of reads that support the same fusion transcript/breakpoint using Expectation Maximization applied to a mixture model. Given a breakpoint formed by joining position s in chromosome X to position t in chromosome Y, we can write the likelihood of alignment A given breakpoint $B = (s, t)$ as:

$$P(A|B) \propto \begin{cases} \mathcal{N}(s - x + t - y|\mu, \sigma) & x \leq s \text{ and } y \leq t \\ 0 & x > s \text{ or } y > t \end{cases}$$

Normalization constant W can be calculated exactly by noting that the volume under the surface defined by $P(A|B)$ is equivalent to an extrusion of the normal distribution by μ, thus $W = \mu$. The assumption that $P(A|B) = 0$ for $x > s$ or $y > t$ implies that the partial alignment process is perfect. We soften this assumption because a distribution with support over the full 2d space of alignment positions will be more conducive to an EM type algorithm. Thus define the soft boundary likelihood as follows:

$$P(A|B) \propto \mathcal{N}(s - x + t - y|\mu, \sigma^2) \cdot e^{-\lambda R(x-s)} \cdot e^{-\lambda R(y-t)}$$

Where $R(x)$ denotes the ramp function defined as follows:

$$R(x) \quad = \quad \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$$

As an approximation, we continue to use the normalization constant $W = \mu$ for the soft boundary likelihood.

Suppose now that the N paired end alignments in $\mathcal{A}$ are produced by a mixture of K breakpoints $\mathcal{B}$. At this stage K is assumed given, below we describe how we determine K. Let $z_{nk} = 1$ if and only if alignment A_n was generated by breakpoint $B_k \in \mathcal{B}$, and let $\pi_k = P(z_{nk} = 1)$. Write the log likelihood of $\mathcal{A}$ given $\mathcal{B}$ as follows:

$$\log p(\mathcal{A}|\mathcal{B}) \quad = \quad \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k P(A_n, z_{nk}|B_k)$$

We use EM to infer the $\mathcal{B}$, $\boldsymbol{\pi}$, and $\boldsymbol{Z}$ that maximize $\log p(\mathcal{A}|\mathcal{B})$. Finally we select K using the Bayesian Information Criterion. We start by setting $K = 1$, running EM, and calculating the BIC (Equation 3). Next we select the paired end alignment m with the minimum model probability (Equation 4). We increment K and augment the previous set of responsibilities, allowing the new cluster to take full responsibility for A_m. We redo EM, starting with the M Step, and calculate BIC for the result. The process continues until the new BIC is larger than the previous, at which point we stop iteration and select the previous solution. Lastly, we assign each paired end alignment n to the cluster k for which $P(z_{nk}|A_n, \hat{\boldsymbol{\pi}}, \mathcal{B})$ is maximum.

$$\text{BIC} = -2 \log p(\mathcal{A}|\mathcal{B}) + 2K \log N \tag{3}$$

$$m = \underset{n}{\operatorname{argmin}}\, p(A'|\mathcal{B}) = \underset{n}{\operatorname{argmin}} \sum_{k=1}^{K} \pi_k p(A_n|B_k) \tag{4}$$

Corroboration between fusion transcripts and breakpoints

Corroboration between fusion transcripts and breakpoints differs significantly between nFuse and Comrad. For Comrad, fusion transcripts and rearrangements breakpoints are said to corroborate given satisfaction of two conditions that loosely determined whether a fusion transcript conceivable arose from a single breakpoint. Corroboration was determined by comparing all fusion transcripts with all breakpoints, and assessing satisfaction of the two conditions. By contrast, nFuse constructs a breakpoint graph from breakpoint predictions, including those supported by ambiguous WGSS reads. nFuse then searches for the shortest alternating path through the breakpoint graph that would corroborate each fusion transcript. As stated above, we add vertices representing the predicted fusion boundaries, and search for

the shortest alternating path between those vertices. If a path exists below a given threshold score, that path is said to corroborate the fusion transcript. nFuse also searches for evidence of CCBRs associated with fusion transcripts. For each breakpoint in an alternating path corroborating a fusion transcript, nFuse searches for a CCBR that includes that breakpoint. As stated above, we first remove the breakpoint edge from the graph, then search for the shortest alternating path between the two vertices of that breakpoint. We also identify fusion transcripts with mapped genomic distance less than 200 kbp, and with an orientation suggestive of a deletion, and label these as read-throughs.

Maximum parsimony formulation for resolving multi-map reads

Let B be the set of all breakpoint predictions implied by WGSS reads G, and let F be the set of all fusion transcript predictions implied by RNA-seq reads R. Let $\mathcal{M}_G \subseteq G \times B$ be a one-to-many mapping from WGSS reads to breakpoints, and let $\mathcal{M}_R \subseteq R \times F$ be a one-to-many mapping from RNA-seq reads to fusion transcripts, each produced by the alignment and clustering process. We would like to identify $\mathcal{U}_G \subseteq \mathcal{M}_G$ and $\mathcal{U}_R \subseteq \mathcal{M}_R$, such that $\mathcal{U}_G$ and $\mathcal{U}_R$ are one-to-one (unique mappings), and such that $\mathcal{U}_G$ and $\mathcal{U}_R$ maximize parsimony. Under the assumption that fusion transcripts and breakpoints are rare, we define the $\mathcal{U}_G$ and $\mathcal{U}_R$ that maximizes parsimony as the $\mathcal{U}_G$ and $\mathcal{U}_R$ that minimizes the number of fusion transcripts and breakpoints. However, we would also like to maximize the number of CGRs we discover, without over-predicting CGRs. We do so by searching within the space of maximum parsimony solutions for a solution that also maximizes the weighted sum of discovered CGRs (herein referred to as the *CGR corroboration score*). Our formulation can thus be seen as a type of multi-objective optimization, for which the minimization of fusion transcripts and breakpoints is primary, and the maximization of the CGR corroboration score is secondary.

We define the space of maximum parsimony solutions as follows. Let $\delta_f(\mathcal{U}_R)$ indicate that $\mathcal{U}_R$ maps at least one RNA-seq read to fusion transcript f. Similarly, Let $\delta_b(\mathcal{U}_G)$ indicate that $\mathcal{U}_G$ maps at least one WGSS read to breakpoint b. The space of maximum parsimony solutions $\mathcal{P}$ is defined as given in Equation 5.

$$\begin{aligned} \mathcal{P}_R &= \underset{\mathcal{U}_R}{\operatorname{argmin}} \sum_f \delta_f(\mathcal{U}_R) \\ \mathcal{P}_G &= \underset{\mathcal{U}_G}{\operatorname{argmin}} \sum_b \delta_b(\mathcal{U}_G) \\ \mathcal{P} &= \mathcal{P}_R \times \mathcal{P}_G \end{aligned} \tag{5}$$

Next we define the CGR corroboration score for paths as follows. Let Q be the set of pairs of fusion transcripts and corroborating paths. Let $\delta_p(\mathcal{U}_G)$ indicate that $\delta_b(\mathcal{U}_G) = 1$ for all breakpoints b in path p. Read-throughs are represented by empty paths and by definition $\delta_p(\mathcal{U}_G) = 1$ for read-throughs. The CGR corroboration score s_p for paths is given by Equation 6.

$$\begin{aligned} s_p(\mathcal{U}_R,\mathcal{U}_G) &= \sum_{(f,p)\in Q} w_p \cdot \delta_f(\mathcal{U}_R) \cdot \delta_p(\mathcal{U}_G) \\ &= \sum_{p} w_p \cdot \delta_p(\mathcal{U}_G) \sum_{(f,p)\in Q} \delta_f(\mathcal{U}_R) \end{aligned} \tag{6}$$

We down-weight more complex paths by defining w_p as given by Equation 7.

$$w_p = \frac{1}{1+|p|} \tag{7}$$

Finally we define the CGR corroboration score for cycles as follows. Let c be a cycle and let $\delta_c(\mathcal{U})$ indicate that $\delta_b(\mathcal{U}) = 1$ for all breakpoints b in cycle c. The CGR corroboration score s_c for cycles is given by Equation 8.

$$s_c(\mathcal{U}_G) = \sum_{c} w_c \ \delta_c(\mathcal{U}_G) \tag{8}$$

Similar to paths, we down-weight more complex cycles by defining w_c as given by Equation 9.

$$w_c = \frac{1}{1+|c|} \tag{9}$$

Our final optimization problem is to identify a solution to Equation 10.

$$\underset{(\mathcal{U}_R,\mathcal{U}_G)\in\mathcal{P}}{\operatorname{argmax}} \left[s_p(\mathcal{U}_R,\mathcal{U}_G) + s_c(\mathcal{U}_G)\right] \tag{10}$$

A complete search of $\mathcal{P}$ would be prohibitively expensive for even small instances of the problem. Thus, we identify an optimal $(\mathcal{U}_R,\mathcal{U}_G)$ using a heuristic search algorithm. Our heuristic is based on the greedy set cover algorithm, and as such guarantees a worst case approximation ratio of $\mathcal{O}(\log n)$ for the problem of minimizing the number of fusion transcripts and breakpoints. No guarantee is provided for maximizing the CGR corroboration score.

The algorithm we propose alternates between minimizing the number of fusion transcripts and minimizing the number of breakpoints. The number of fusion transcripts is minimized, and the CGR corroboration score maximized, given an estimate for the value of δ_p. Next the number of breakpoints is minimized, and the CGR corroboration score maximized, given an estimate for the value of $\kappa_p = \sum_{(f,p)\in Q} \delta_f$. The value κ_p represents an estimate of the number of fusion transcripts corroborated by path p. At each iteration we re-estimate values for δ_p and κ_p.

For path p, let $\chi_p = \left(\delta_{p_1}, \delta_{p_2}, ..., \delta_{p_i}\right)$ be a sequence of δ_p values from previous solutions to the problem of minimizing breakpoints. We initialize χ_p to a sequence of m successes, placing

a heavy prior on existence of each path in the form of m pseudo-counts. Furthermore, let $\psi_p = \left(\kappa_{p_1}, \kappa_{p_2}, ..., \kappa_{p_i}\right)$ be a sequence of κ_p values from previous solutions to the problem of minimizing fusion transcripts. We initialize ψ_p as follows. For each path p, identify $\text{ub}(\kappa_p)$, an upper bound on κ_p, by running Algorithm 3 (with null bonuses) for only the fusion transcripts in the set $\{f : (f, p) \in Q\}$. Initialize ψ_p as a sequence of m values $\text{ub}(\kappa_p)$, placing a heavy prior on the existence of the fusion transcripts corroborating p.

The algorithm proceeds as follows. Let Z be the set of complex rearrangements (paths and cycles).

1. Estimate $\hat{\kappa}_p = \frac{\sum_i \kappa_{p_i}}{|\psi_p|}$. Let V be a set of bonuses for rearrangements, where v_z is a bonus given for the solution that selects rearrangement z. Set the bonus v_c for cycle c as w_c and the bonus v_p for path p as $\hat{\kappa}_p w_p$. Use Algorithm 1 to select a minimal set of breakpoints while attaining the maximum total in bonuses.
   Let $C_G = \text{MinimizeBreakpoints}(G, B, Z, V, \epsilon)$. Create $\mathcal{U}_G$ from C_G by assigning each read g to the breakpoint in C_G that covers g and contains the greatest number of reads, breaking ties randomly.

2. For each path p, calculate δ_p based on $\mathcal{U}_G$ from step 1, and append it to χ_p.

3. Estimate $\hat{\delta}_p = \frac{\sum_i \delta_{p_i}}{|\chi_p|}$. Let Y be a set of bonuses for fusion transcripts, where y_f is a bonus for a solution that selects fusion transcript f. Set the bonus y_f for fusion transcript f corroborated by path p as $\hat{\delta}_p w_p$. If f is a read-through then $y_f = 1$ since $\delta_p = 1$ and $w_p = 1$ by definition for read-throughs. If f is neither a read-through or corroborated by a path, $y_f = 0$. Use Algorithm 3 to select a minimal set of fusion transcripts while attaining the maximum total in bonuses.
   Let $C_R = \text{MinimizeFusions}(R, F, Y, \epsilon)$. Create $\mathcal{U}_R$ from C_R by assigning each read r to the fusion transcript in C_R that covers r and contains the greatest number of reads, breaking ties randomly.

4. For each path p, calculate κ_p based on $\mathcal{U}_R$ from step 3, and append it to ψ_p.

5. Repeat n times and return $\mathcal{U}_G$ and $\mathcal{U}_R$ from the most recent iteration.

For the purposes of this study, we have used $m = 10$ and $n = 10$.

Algorithm 1 takes as input B, Z and V. Each element of B is a labelled set of reads representing a breakpoint. Each element of Z is a set of labels representing a set of breakpoints putatively forming a CGR. Y contains a bonus for each CGR in Z. Initially, each breakpoint is given a weight 1. The bonus v_z for CGR z is distributed evenly between each breakpoint $b \in z$ by subtracting a small bonus, $\epsilon \frac{v_z}{|z|}$ from the weight of each b. At each iteration the algorithm selects the most cost efficient breakpoint b' and adds it to the set of chosen breakpoints C. All breakpoints not in C but completely covered by $\bigcup C$ are considered *invalid*. All CGRs containing invalid breakpoints are invalid. The bonuses of invalid CGRs are removed from the CGR's remaining valid breakpoints. Finally, for each CGR z containing b', the bonus from z is redistributed to the remaining breakpoints in z that are not in C. The algorithm terminates when C covers all reads. Calculation of weights is given by Algorithm 2.

Algorithm 1 MinimizeBreakpoints(G,B,Z,V,ϵ)

INPUT: WGSS reads G
INPUT: Breakpoints B as labelled sets of reads
INPUT: CGRs Z as sets of breakpoints
INPUT: CGRs bonus V
INPUT: Small constant ϵ
OUTPUT: Breakpoints $C \subseteq B$

```
C ← ∅
for all b ∈ B do
  w_b ← CalculateBreakpointWeight(b,B,Z,V,C,ε)
end for
while ⋃C ≠ G do
  select b ∉ B that minimizes w_b / |b \ ⋃C|
  C ← C ∪ {b}
  for all b' ∈ B : b' ≠ b, b' ⊆ ⋃C do
    for all z ∈ Z : b' ∈ z do
      Z ← Z \ {z}
      for all b'' ∈ z do
        w_b'' ← CalculateBreakpointWeight(b'',B,Z,V,C,ε)
      end for
    end for
    B ← B \ {b'}
  end for
  for all z ∈ Z : b ∈ z do
    for all b' ∈ z do
      w_b' ← CalculateBreakpointWeight(b',B,Z,V,C,ε)
    end for
  end for
end while
return C
```

Algorithm 2 CalculateBreakpointWeight(b,B,Z,V,C,ϵ)

INPUT: Breakpoint b
INPUT: Valid breakpoints B as labelled sets of reads
INPUT: Valid CGRs Z as sets of breakpoints
INPUT: CGR bonuses V
INPUT: Chosen breakpoints C
INPUT: Small constant ϵ
OUTPUT: weight w

$w \leftarrow 1$
for all $z \in Z$ **do**
  if $z \subseteq B$ **then**
    for all $b \in z$ **do**
      $u = b \setminus \mathcal{C}$
      $w = w - \frac{\epsilon \cdot v_b}{|u|}$
    end for
  end if
end for
return w

Algorithm 3 takes as input F and Y. Each element of F is a labelled set of reads representing a fusion transcript. Y contains a bonus for each fusion transcript in F. Each fusion transcript f is given a weight $1-\epsilon \cdot y_f$. At each iteration the algorithm selects the most cost efficient breakpoint f' and adds it to the set of chosen breakpoints C. The algorithm terminates when C covers all reads.

Algorithm 3 MinimizeFusions(R,F,Y,ϵ)

INPUT: RNA-seq reads R
INPUT: Fusion transcripts F as labelled sets of reads
INPUT: Fusion transcripts bonuses Y
INPUT: Small constant ϵ
OUTPUT: Fusion transcripts $C \subseteq F$

$C \leftarrow \emptyset$
for all $f \in F$ **do**
  $w_f \leftarrow 1 - \epsilon \cdot y_f$
end for
while $\bigcup C \neq R$ **do**
  select $f \notin F$ that minimizes $\frac{w_f}{|f \setminus \bigcup C|}$
  $C \leftarrow C \cup \{f\}$
end while
return C

Post-processing

The result of the mult-map resolution stage of Comrad will be a set of fusion transcripts and corroborating breakpoints. Assembled sequences for breakpoint predictions are re-aligned to the reference genome using blat. Breakpoint sequences that align with 90% identity within 2kb sized genomic region are filtered, as are the associated CGRs. The fusion transcripts are further processed as for deFuse (McPherson et al., 2011a). In brief, targeted dynamic programming is used to identify split reads and assemble nucleotide level fusion transcripts. A probability is calculated for each assembled fusion transcript using a classifier trained on known positive and negative fusion transcript predictions.

Calculating breakpoint probability

We predict breakpoints from discordant paired end alignments. Our approach aims for high sensitivity by including reads with multiple genomic mappings, and reads that map only partially to the genome. To ensure adequate specificity, we calculate a probability for each breakpoint based on the alignment evidence and use that probability in downstream analysis including CGR discovery.

Let $\mathbf{R}$ be the set of paired end WGSS reads. We generate a set of mapping locations $\mathbf{M}$ for $\mathbf{R}$ using the following well established strategy (Tuzun et al., 2005; Volik et al., 2003). For each paired end read $(r_j^1, r_j^2) \in \mathbf{R}$:

1. identify a single concordant mapping location if it exists.
2. if no concordant mapping location exists:
   (a) identify the n top scoring mapping locations for r_j^1
   (b) identify the n top scoring mapping locations for r_j^2

We identify the n top scoring mapping locations for r_j^1 (and r_j^2) as follows. Let s_j be the maximum alignment score attained by *partial* alignment of read j to the genome. Let k be the number of mappings of read j that attain s_j. If $k > n$ assume the read is unmappable and filter it, otherwise retain the k mapping locations.

Let $\mathbf{m}_j \in \mathbf{M}$ be the mapping locations identified for read $(r_j^1, r_j^2) \in \mathbf{R}$. Define the following indicator variables:

$$\begin{aligned} c_j &\equiv \text{read } j \text{ is concordant} \\ d_j &\equiv \text{the true alignment was discovered and is in the set } \mathbf{m}_j \end{aligned}$$

We make the assumption that reads mapped concordantly by the aligner are in fact concordant (with probability 1). We filter the concordantly mapped reads to create the set of discordant reads $\mathbf{R}^d$ and set of discordant mappings $\mathbf{M}^d$. As a result, $P(c_j = 1, d_j = 1) = 0$ for the set of filtered reads. We estimate probabilities for the remaining two possibilities for the true alignment of each read:

$$\begin{aligned} P(c_j = 1|\cdot) &\equiv \text{concordant but missed by the aligner} \\ P(d_j = 1|c_j = 0, \cdot) &\equiv \text{discordant but missed by the aligner} \end{aligned}$$

We estimate $P(c_j = 1|\cdot)$ using the maximum concordant alignment score cs_j. To calculate cs_j, we align both ends of read j to all mapping locations in the set $\mathbf{m}_j$, and set cs_j to the maximum alignment score identified by this process. We then calculate $P(c_j = 1|cs_j)$, and use it to approximate $P(c_j = 1|\cdot)$. We approximate $P(d_j = 1|c_j = 0, \cdot)$ as $P(d_j = 1|c_j = 0, as_j)$ where as_j is the alignment score for read j.

Next, we cluster the discordant alignments $\mathbf{M}^d$ based on the likelihood that a set of alignments were generated by the same breakpoint. Let the resulting clusters of alignments represent putative breakpoints. Let g_{ij} indicate that putative breakpoint i generated read j. Assume $g_{ij} = 0$ if read j is not in the cluster that supports breakpoint i. We estimate $P(g_{ij} = 1|\cdot)$ as $P(g_{ij} = 1|nm_j, d_j = 1)$, where nm_j is the number of alternate mapping locations of read j. Under the assumption that all mapping locations discovered by the aligner are equally likely, we calculate $P(g_{ij} = 1|nm_j, d_j = 1) = \frac{1}{nm_j}$.

Finally, let b_i indicate that breakpoint i is true, let $\mathbf{G}_i$ be the set of all g_{ij} for breakpoint i, and let n_i be the number of reads that were generated by breakpoint i, that is $n_i = \sum_{g_{ij} \in \mathbf{G}_i} g_{ij}$. We estimate $P(b_i|n_i)$ and use it to estimate $P(b_i|\cdot)$ as given by equation 11.

$$\begin{aligned} P(b_i|\cdot) = \sum_{\mathbf{G}_i} P(b_i|n_i) \prod_j & P(g_{ij} = 1|nm_j, d_j = 1) \\ & \times P(d_j = 1|as_j, c_j = 0) \\ & \times P(c_j = 0|cs_j) \end{aligned} \tag{11}$$

We first describe methods for estimating the above probability distributions, then describe an algorithm for calculating $P(b_i|\cdot)$ from these distributions.

Calculating $P(c_j = 0|cs_j)$

Alignment scores are calculated using dynamic programming with affine gap penalties (Gotoh, 1982). We use the scoring function implemented for bowtie2 (Langmead and Salzberg, 2012), redescribed here. A gap of length N is given a penalty calculated as,

$$\text{GO} + N \times \text{GE}.$$

We use the bowtie2 defaults, GO = 5 and GE = 3. Mismatches are given a penalty calculated as,

$$\text{MN} + \text{floor}\left(\frac{\text{MX} - \text{MN}}{\frac{1}{40}\min(Q, 40.0)}\right)$$

where Q is the Phred quality value. We use the bowtie2 defaults, MN = 2 and MX = 6. Matches are not given a bonus.

We calculate the maximum concordant alignment score cs_j for discordant read j as follows. Let r_j^1 and r_j^2 be end 1 and 2 of read j, and let $\mathbf{m}_j^1$ and $\mathbf{m}_j^2$ be the discordant mappings of each end. Mappings $\mathbf{m}_j^1$ and $\mathbf{m}_j^2$ were generated by a partial alignment of r_j^1 and r_j^2. First truncate r_j^1 and r_j^2 to the maximum aligned lengths in $\mathbf{m}_j^1$ and $\mathbf{m}_j^2$ respectively, to form tr_j^1 and tr_j^2 respectively. For each mapping $m_{jk}^1 \in \mathbf{m}_j^1$ align tr_j^2 to the 1000nt region adjacent to m_{jk}^1 in the genome (downstream if the strand of m_{jk}^1 is "+", upstream if the

strand of m^1_{jk} is "-"). Repeat for $m^2_{jk} \in \mathbf{m}^2_j$ and tr^1_j. Calculate cs_j as the maximum of all scores identified using the above procedure.

Letting $P(c_j = 0) = \pi_c$, we can calculate $P(c_j = 0|cs_j)$ using bayes rule.

$$P(c_j = 0|cs_j) = \frac{P(cs_j|c_j = 0)\pi_c}{P(cs_j|c_j = 0)\pi_c + P(cs_j|c_j = 1)(1 - \pi_c)}$$

We estimate $P(cs_j|c_j = 0)$ from alignments of reads to random locations in the genome. We first uniformly sample 1/1000 reads from the WGSS data. For each sampled read, we produce copies of the read truncated to lengths ranging from 20nt to the length of the read. We then align the truncated reads to genomic locations selected uniformly and at random. For each truncation length, we use the samples to calculate a density using gaussian kernel density estimation with bandwidth 1. Let f^ℓ be the resulting density for truncation length ℓ. Since cs_j is the result of multiple trials, we cannot naively use f^ℓ to calculate $P(cs_j|c_j = 0)$. We instead calculate extreme value distributions based on f^ℓ, and use these to calculate $P(cs_j|c_j = 0)$. Let t be the number of trials used to calculate maximum concordant alignment score x, and let $f^\ell_t(x)$ represent to probability of attaining maximum concordant alignment score cs_j after t trials. $f^\ell_t(cs_j)$ can be calculated from $f^\ell_1 = f^\ell$, and the cumulative density F^ℓ_1 using the following recursion.

$$f^\ell_t = f^\ell_{t-1} \cdot F^\ell_1 + F^\ell_{t-1} \cdot f^\ell_1 - f^\ell_{t-1} \cdot f^\ell_1$$

We then estimate $P(cs_j|c_j = 0) = f^\ell_t(cs_j)$.

We estimate $P(cs_j|c_j = 1)$ from concordant alignments. We first uniformly sample 1/1000 reads from the WGSS data. For each sampled read, we produce copies of the read truncated to lengths ranging from 20nt to the length of the read. We then align the truncated reads to genomic locations given by the concordant alignments of the original non-truncated reads. Given that read j is concordant, the alignment score cs_j represents the score produced by aligning read j to its single location of origin. Thus for $P(cs_j|c_j = 1)$ we do not use an extreme value distribution. Instead, for each truncation length we fit a negative binomial distribution to the samples and use that to estimate $P(cs_j|c_j = 1)$.

Finally, we estimate π_c using the EM algorithm. Let J be the number of potentially discordant reads. The expected value of the log likelihood function with respect to the conditional distribution of all $c_j \in \mathbf{C}$ given $\pi^{(t)}_c$ and $cs_j \in \mathbf{S}$ is as follows.

$$\begin{aligned} E_{\mathbf{C}|\mathbf{S},\pi^{(t)}_c}[\log L(\pi_c|\mathbf{C},\mathbf{S})] &= \sum_j (1 - P(c_j = 0|cs_j, \pi^{(t)}_c)) \cdot \log[(1 - \pi_c) \cdot P(cs_j|c_j = 1)] \\ &\quad + P(c_j = 0|cs_j, \pi^{(t)}_c) \cdot \log[\pi_c \cdot P(cs_j|c_j = 0)] \end{aligned}$$

The maximum likelihood estimates of π_c yield the following update equation.

$$\pi^{(t+1)}_c = \frac{\sum_k P(c_j = 0|cs_j, \pi^{(t)}_c)}{J}$$

Thus we calculate $P(c_j = 0|cs_j)$ and π_c by iteratively calculating $P(c_j = 0|cs_j, \pi^{(t)}_c)$ for all j (Expectation step) and using those values to calculate $\pi^{(t+1)}_c$ (Maximization step) repeating until convergence.

Calculating $P(d_j = 1|as_j, c_j = 0)$

Alignment scores as_j are calculated using dynamic programming with affine gap penalties as described in the previous section. A discordant read may have a marginal alignment score for the following reasons:

i the read is poor quality
ii the region has polymorphisms compared to the reference genome
iii the read is non-genomic
iv the read is mapped to the wrong location

We would like to distinguish the first two possibilities from the last two possibilities. We do so by calculating $P(d_j = 1|as_j, c_j = 0)$, the probability that discordant read j has a valid discordant mapping given its alignment score (referred to herein as simply $P(d_j = 1|as_j)$. We can estimate $P(as_j|d_j = 1)$ from concordant alignments similar to how we estimated $P(cs_j|c_j = 1)$ in the previous section. Unfortunately it is difficult to estimate $P(as_j|d_j = 0)$. Thus we formulate the problem of calculating $P(d_j = 1|as_j)$ as a learning problem with positive and unlabelled data and use a modified version of a previously described method (Elkan and Noto, 2008). Let indicator s_j represent whether read j has been *sampled*. Then we can write,

$$P(d_j = 1|as_j) \quad = \quad \frac{P(s_j = 1|as_j)}{P(s_j = 1|d_j = 1)}$$

We first sample m scores from the concordant alignments (see previous section), where m is the number of discordant alignments. We then build a k-nearest-neighbour classifier ($k = 0.05m$) from the *unlabelled* scores U from the discordant alignments, and the *labelled positive* scores L from the concordant alignments. The KNN classifier will yield a function $g(as_j) = P(s_j = 1|as_j)$. We then estimate $c = P(s_j = 1|d_j = 1)$ from the labelled positive scores as described previously (estimator 1 from Elkan and Noto (2008)).

$$\hat{c} \quad = \quad \frac{1}{|L|} \sum_{as \in L} g(as)$$

Calculating $P(g_{ij} = 1|nm_j, d_j = 1)$

We assume all discordant alignment mapping locations are equally likely. Thus,

$$P(g_{ij} = 1|nm_j, d_j = 1) \quad = \quad \frac{1}{nm_j}.$$

Calculating $P(b_i|n_i)$

Calculation of $P(b_i|n_i)$ is formulated as a positive unlabelled learning problem and uses a similar technique as described for calculating $P(d_j = 1|as_j, c_j = 0)$. We build a set of *labelled positives* as follows. Select 100,000 genomic positions uniformly at random, and identify the number of reads that span (spanning read count) those positions from concordant alignments. Remove positions not covered by any reads and re-sample to produce a set of

m spanning read counts, where m is the number of breakpoint predictions. Build a k-nearest neighbour classifier ($k = 0.05m$) from the *labelled positive* spanning read counts from concordant alignments and the *unlabelled* spanning read counts from the breakpoint predictions. Estimate $P(b_i|n_i)$ as described above for $P(d_j = 1|as_j, c_j = 0)$.

Calculating $P(b_i|\cdot)$

Finally, we require an efficient way of calculating a sum over all possible settings of the values of $\mathbf{G}_i$, and we do this using dynamic programming similar to previously described methods (Hormozdiari et al., 2009). Rearranging the expression for $P(b_i|\cdot)$ we obtain the following.

$$P(b_i|\cdot) = \sum_{n_i=0}^{|\mathbf{G}_i|} P(b_i|n_i) \sum_{\mathbf{G}_i:\sum_j g_{ij}=n_i} \prod_j P(g_{ij} = 1|nm_j, as_j, cs_j)$$

Let $f(p, q)$ be defined as follows.

$$f(p, q) = \sum_{\mathbf{G}_i:\sum_j g_{ij}=p} \prod_{j=1}^{q} P(g_{ij}|nm_j, as_j, cs_j)$$

Then we can calculate $f(\{0, 1, ..., |\mathbf{G}_i|\}, |\mathbf{G}_i|)$ in $O(n^2)$ time using the following recurrence.

$$\begin{aligned} f(0, 0) &= 1 \\ f(-1, q) &= 0 \\ f(q + 1, q) &= 0 \\ f(p, q) &= f(p - 1, q - 1) \cdot P(g_{ij} = 1|nm_j, as_j, cs_j) + \\ &\quad f(p, q - 1) \cdot P(g_{ij} = 0|nm_j, as_j, cs_j) \end{aligned}$$

Given $f(\{0, 1, ..., |\mathbf{G}_i|\}, |\mathbf{G}_i|)$, we can calculate $P(b_i|\cdot)$ as follows.

$$P(b_i|\cdot) = \sum_{n_i=0}^{|\mathbf{G}_i|} P(b_i|n_i) \cdot f(n_i, |\mathbf{G}_i|)$$

Shortest alternating path algorithm

The algorithm we propose for finding the shortest alternating path follows the algorithm proposed by Brown (1974). We first create a new directed graph H from the undirected breakpoint graph G as follows. For each vertex v in the G, add two vertices, v_{in} and v_{out} in H. If (v_1, v_2) is a breakpoint edge in the G, add the edge (v_{1in}, v_{2out}) to H. If (v_1, v_2) is an adjacency edge in the G, add the edge (v_{1out}, v_{2in}) to H. We then find a shortest alternating path in G by finding a shortest path in H.

Finding the shortest path in H is a non-trivial problem. Adjacency edges connect vertex v in G to vertices that are: a) on the same chromosome as v, and b) and have opposite direction to v. Thus G and also H can be considered dense. The algorithm we propose is similar to dijkstras algorithm, however it has the benefit of being faster than dijkstras for a constrained search in a dense graph. We constrain our shortest path search in 2 ways. First, we search for paths with length less than a maximum allowable distance. Second, we place a limit on the number of vertices relaxed by the algorithm. Like dijkstras we maintain a set of vertices S for which we know the shortest path from the starting vertex s to any vertex in S. Also like dijkstras, we maintain a priority queue of the neighbours of S that are closest to vertices in S. However, we refrain from maintaining a priority queue containing *all* vertices adjacent to S. Instead, we maintain a priority queue such that for each vertex v in the priority queue, v is the vertex in $\bar{S}$ that is the closest neighbor in $\bar{S}$ of some vertex in S. We also maintain, for each vertex v in the priority queue, a set, $p(v)$ of the vertices in S for which v is the closest neighbor in $\bar{S}$.

Clearly, the vertex v_{next} at the top of the priority queue will be the same for our algorithm as for dijkstras. We relax v_{next} by adding it to S and recording the shortest path to v_{next}. We then maintain the priority queue by doing the following. For each vertex u in the set $\{v_{next}\} \cup p(v_{next})$, find the vertex v in $\bar{S}$ that is closest to u, and add v to the priority queue if it exists. For this task we require a list of the neighbours of u sorted by their distance from u. If the outgoing edges from u in H are breakpoint edges in G this is trivial, since there is only one outgoing edge. If the outgoing edges from u in H are adjacency edges in G, then we can simply use a sorted list of breakpoint positions, where the same sorted list need not be duplicated multiple times for different vertices.

Suppose that we wish to constrain our search to a maximum of k relaxation steps. At each relaxation step we select the next closest vertex, v_{next}, an $O(\log k)$ operation that will be repeated k times. Next we iterate through the set $p(v_{next})$, finding the next closest vertices. In the worst case, at the k^{th} step, all $k-1$ vertices in S will be in the set $p(v_{next})$. Furthermore, for each of these $k-1$ vertices the next closest $k-2$ vertices will be in S. Thus for each of the $k-1$ vertices in S, the algorithm will iterate over the the other $k-2$ vertices before getting to the vertex in $\bar{S}$ that is closest. The complexity of the algorithm is dominated by this step and is worst case k^2. Thus the algorithm will be beneficial if $k \ll n$, where n is the number of vertices, since dijkstras would use best case $O(kn)$ operations.

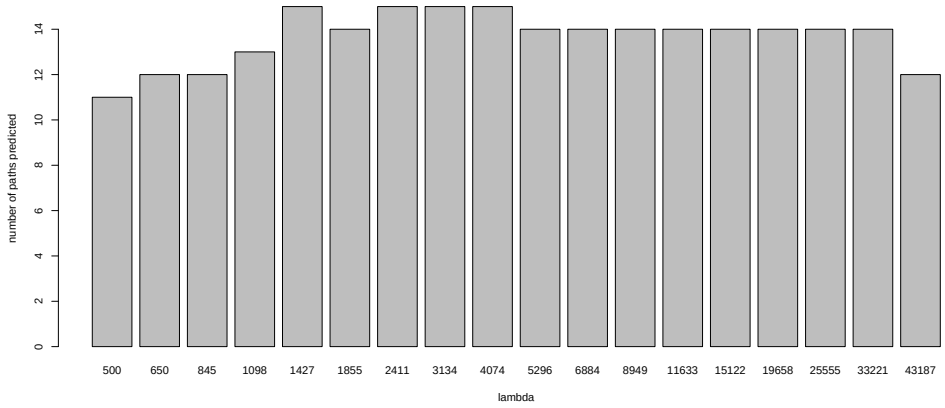
Path search parameter β_p

The parameter β_p can be seen as a mixing parameter for the two types of edges in the shortest alternating path optimization problem. Lower values of lambda will place more importance on the genomic distances between breakpoints in a path (adjacency edges), whereas higher values will place more importance on breakpoint probabilities.

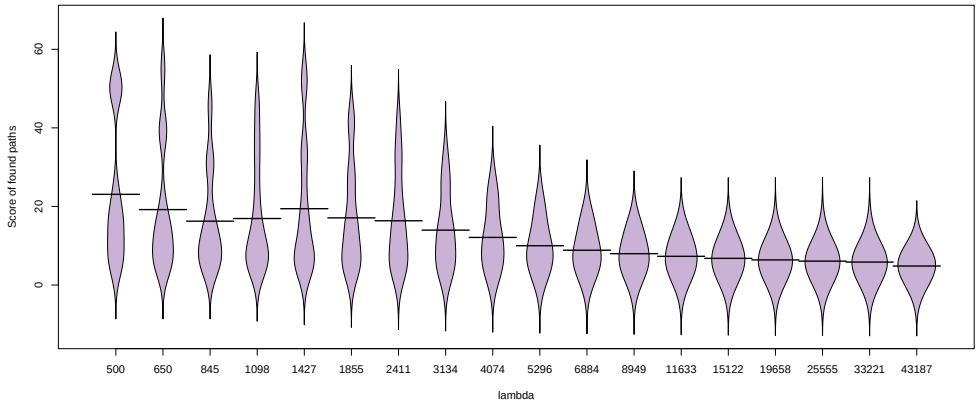
We sought to estimate the effect of β_p on path searches using the HCC1954 data. A range of β_p values was selected, and the shortest alternating path algorithm was used to predict paths for each value of β_p. We made the strong assumption that paths were *valid* if they were composed entirely of breakpoints validated in 3 previous studies (Bignell et al., 2007; Stephens et al., 2009; Galante et al., 2011).

We then used the resulting *valid* paths to estimate an optimal value of β_p, and measure

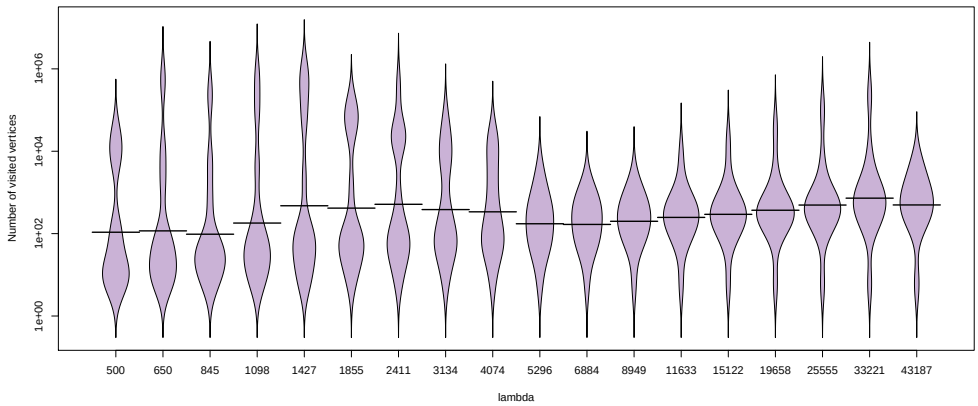
the effect of β_p on the shortest path algorithm. The number of valid paths ranged from 11 to 14 across the range of β_p values (Figure 8a), with the greatest number predicted between 1427 and 4074. As expected, scores for the valid breakpoints decreased almost monotonically with β_p (Figure 8b). We also analyzed the number of vertices visited by the algorithm (visit count) for each valid path, as a proxy for the time spent on the search. Interestingly, the distribution of visit counts for valid paths was bimodel for lower values of β_p, indicating that some proportion of the paths would take significantly longer to be identified at these β_p values. At a value of $\beta_p = 5296$, the distribution of visit counts becomes unimodel, and is at a local minima. We selected $\beta_p = 6884$, maximum visit count of 300,000, and maximum score of 30 based, as an adequate balance of sensitivity and balancing sensitivity and running time. Interestingly, the value we selected for β_p is close to the estimate of 6082 gained by fitting the distribution of intron lengths to an exponential distribution.



(a)



(b)



(c)

Figure 8: Statistics for paths composed of previously validated breakpoints (valid paths), as identified using a range of values for lambda. (a) Number of valid paths identified. (b) Distribution of scores for valid paths. (c) Distribution of the number of vertices visited by the algorithm for valid paths.