BIOINFORMATICS

Vol. 00 no. 00 2014
Pages 1–8

Cloudbreak: Accurate and Scalable Genomic Structural Variation Detection in the Cloud with MapReduce

Christopher W. Whelan [1], Jeffrey Tyner [2], Alberto L'Abbate [3], Clelia Tiziana Storlazzi,[3], Lucia Carbone [4], and Kemal Sönmez [1,5,*]*

[1]Institute on Development and Disability and Center for Spoken Language Understanding, Oregon Health & Science University, Portland, OR, USA
[2] Program in Molecular and Cellular Biosciences, Oregon Health & Science University, Portland, OR, USA
[3] Department of Biology, University of Bari "Aldo Moro", Via G. Amendola 165/A, 70126, Bari, Italy
[4] Behavioral Neuroscience Department, Oregon Health & Science University, Portland, OR, USA
[5] Department of Medical Informatics & Clinical Epidemiology, Oregon Health & Science University, Portland, OR, USA

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: The detection of genomic structural variations (SV) remains a difficult challenge in analyzing sequencing data, and the growing size and number of sequenced genomes have rendered SV detection a bona fide big data problem. MapReduce is a proven, scalable solution for distributed computing on huge data sets.

Results: We describe an algorithmic framework for SV detection algorithms in MapReduce based on computing local genomic features, and use it to develop a deletion and insertion detection algorithm, Cloudbreak. On simulated and real data sets, Cloudbreak achieves accuracy improvements over popular SV detection algorithms, and genotypes variants from diploid samples. It provides dramatically shorter runtimes and the ability to scale to big data volumes on large compute clusters. Cloudbreak includes tools to set up and configure MapReduce (Hadoop) clusters on cloud services, enabling on-demand cluster computing.

Availability: Source code, binary releases, example scripts, and a full user manual are available at `http://github.com/cwhelan/cloudbreak`

Contact: sonmezk@ohsu.edu

1 INTRODUCTION

Genomic structural variations (SVs) such as deletions, insertions, and inversions of DNA are widely prevalent in human populations and account for the majority of the bases that differ among normal human genomes (Mills *et al.*, 2011a; Conrad *et al.*, 2010). However, detection of SVs with current high-throughput sequencing technology remains a difficult problem, with limited concordance between available algorithms and high false discovery rates (Mills *et al.*, 2011a). Part of the problem stems from the fact that the signals indicating the presence of SVs are spread throughout large data sets, and integrating them to form an accurate detection measure is computationally difficult. As the volume of massively parallel sequencing data approaches "big data" scales, SV detection is becoming a time consuming component of genomics pipelines, and presents a significant challenge for research groups and clinical operations that may not be able to scale their computational infrastructure. Here we present a distributed software solution that is scalable and readily available on the cloud.

In other fields that have taken on the challenge of handling very large data sets, such as internet search, scalability has been addressed by computing frameworks that distribute processing to many compute nodes, each working on local copies of portions of the data. In particular, Google's MapReduce (Dean and Ghemawat, 2008) framework was designed to manage the storage and efficient processing of very large data sets across clusters of commodity servers. Hadoop is an open source project of the Apache Foundation which provides an implementation of MapReduce. MapReduce and Hadoop allow efficient processing of large data sets by executing tasks on nodes that are as close as possible the data they require, minimizing network traffic and I/O contention. The Hadoop framework has been shown to be effective in sequencing-related tasks including short read mapping (Schatz, 2009), SNP calling (Langmead *et al.*, 2009), and RNA-seq differential expression analysis (Langmead *et al.*, 2010).

Hadoop/MapReduce requires a specific programming model, however, which can make it difficult to design general-purpose algorithms for arbitrary sequencing problems like SV detection. MapReduce divides computation across a cluster into three phases. In the first phase, *mappers* developed by the application programmer examine small blocks of data and emit a set of $\langle key, value\rangle$ pairs for each block examined. The framework then sorts the output of the mappers by key, and aggregates all values that are associated with each key. Finally, the framework executes *reducers*, also created by the application developer, which process all of the values for a particular key and produce one or more outputs that summarize or aggregate those values.

*to whom correspondence should be addressed

Popular SV detection algorithms use three main signals present in high-throughput sequencing data sets (Alkan *et al.*, 2011). Read-pair (RP) based methods use the distance between and orientation of the mappings of paired reads to identify the signatures of SVs (Campbell *et al.*, 2008; Chen *et al.*, 2009; Hormozdiari *et al.*, 2009; Sindi *et al.*, 2009; Korbel *et al.*, 2009). Traditionally, this involves separating mappings into those that are *concordant* or *discordant*, where discordant mappings deviate from the expected insert size or orientation, and then clustering the discordant mappings to find SVs supported by multiple discordantly mapped read pairs. Read-depth (RD) approaches, in contrast, consider the changing depth of coverage of concordantly mapped reads along the genome (Abyzov *et al.*, 2011; Alkan *et al.*, 2009; Yoon *et al.*, 2009; Chiang *et al.*, 2009). Finally, split-read (SR) methods look for breakpoints by mapping portions of individual reads to different genomic locations (Wang *et al.*, 2011; Ye *et al.*, 2009).

Many RP methods consider only unambiguously discordantly mapped read pairs. Some approaches also include ambiguous mappings of discordant read pairs to improve sensitivity in repetitive regions of the genome (Hormozdiari *et al.*, 2009; Quinlan *et al.*, 2010). Several recent RP approaches have also considered concordant read pairs, either to integrate RD signals for improved accuracy (Sindi *et al.*, 2012; Michaelson and Sebat, 2012; Chiara *et al.*, 2012), or to eliminate the thresholds that separate concordant from discordant mappings and thus detect smaller events (Marschall *et al.*, 2012). Increasing the number of read mappings considered, however, increases the computational burden of SV detection.

Our goal is to leverage the strengths of the MapReduce computational framework in order to provide fast, accurate and readily scalable SV-detection pipelines. The main challenge in this endeavor is the need to separate logic into mappers and reducers, which makes it difficult to implement traditional RP-based SV detection approaches in MapReduce, particularly given the global clustering of paired end mappings at the heart of many RP approaches. MapReduce algorithms, by contrast, excel at conducting many independent calculations in parallel. In sequencing applications, for example, MapReduce based SNV-callers Crossbow (Langmead *et al.*, 2009) and GATK (McKenna *et al.*, 2010) perform independent calculations on partitions of the genome. SV approaches that are similarly based on local computations have been described: the RP-based SV callers MoDIL (Lee *et al.*, 2009) and forestSV (Michaelson and Sebat, 2012) compute scores or features along the genome and then produce SV predictions from those features in a post-processing step. We will show that this strategy can be translated into the MapReduce architecture.

In this paper, we describe a framework for solving SV detection problems in Hadoop based on the computation of local genomic features from paired end mappings. In this framework we have developed a software package, Cloudbreak, that discovers genomic deletions up to 25,000bp long, and short insertions. Cloudbreak computes local features based on modeling the distribution of insert sizes at each genomic location as a Gaussian Mixture Model (GMM), an idea first implemented in MoDIL (Lee *et al.*, 2009). The use of Hadoop enables scalable implementations of this class of algorithm. We characterize our algorithm's performance on simulated and real data sets and compare its performance to those of several popular methods.

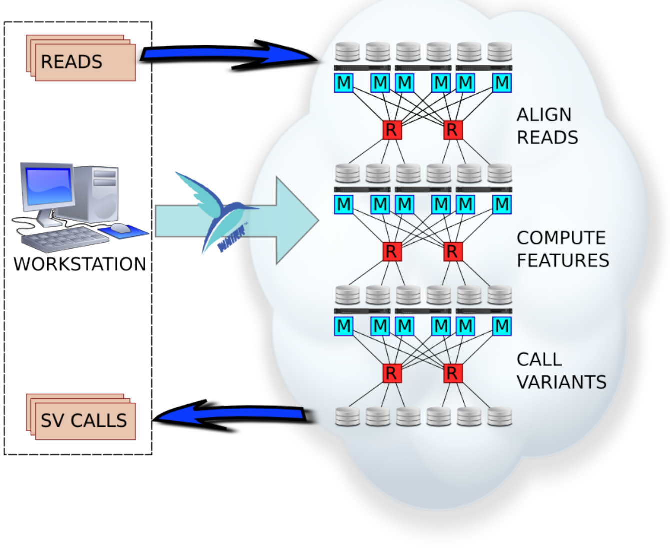

Fig. 1. An overview of the Cloudbreak workflow. Reads are first uploaded to a Hadoop cluster from local storage. Cloudbreak then executes three MapReduce jobs to process the data: 1) Mapping with sensitive settings. 2) Computation of features across the genome. 3) Calling structural variations based on the features computed in the previous step. Finally, SV predictions can be downloaded from the Hadoop cluster and examined. Cloudbreak can also use the Apache Whirr library to automatically provision Hadoop clusters on and deploy data to cloud providers such as Amazon Elastic Compute Cloud.

Finally, and quite importantly from a practical point of view, our implementation of Cloudbreak provides the functionality to easily set up and configure Hadoop clusters on cloud service providers, making dynamically scalable distributed SV detection accessible to all whose computational needs demand it.

2 METHODS

2.1 Cloudbreak: A Hadoop/MapReduce Software Package for SV Detection

Our framework for SV detection in MapReduce divides processing into three distinct MapReduce jobs (Figure 1): a job that can align reads to the reference using a variety of mapping algorithms; a job that computes a set of features along the genome; and a job which calls structural variations based on those features. Our description of a general MapReduce SV detection algorithmic framework and how Cloudbreak is implemented within that framework are provided in Supplementary Algorithm **??** and the Supplementary Materials; here we proceed with a high-level description of the Cloudbreak implementation.

Cloudbreak's alignment job can run a variety of alignment tools that report reads in SAM format (Supplementary Materials). In the map phase, mappers align reads in either single-end or paired-end mode to the reference genome in parallel, outputting mapping locations as values under a key identifying the read pair. In the reduce phase, the framework combines the reported mapping locations for the two ends of each read pair. This job can also be skipped in favor of importing a pre-existing set of mappings directly into the Hadoop cluster.

In the next job, Cloudbreak computes a set of features for each location in the genome. To begin, we tile the genome with small fixed-width, non-overlapping windows. For the experiments reported in this paper we use a window size of 25bp. Within each window, we examine the distribution of insert sizes of mappings that span that window, and compute features by fitting a GMM to that distribution (Supplementary Materials, Supplementary Fig. **??**). To remove incorrect mappings, we use an adaptive quality cutoff for each genomic location and then perform an outlier-detection based noise reduction technique (Supplementary Materials); these procedures also allow us to process multiple mappings for each read if they are reported by the aligner.

Finally, the third MapReduce job is responsible for making SV calls based on these features. In this job, we search for contiguous blocks of genomic locations with similar features and merge them into individual insertion and deletion SV calls after applying noise reduction (Supplementary Materials). Reducers process each chromosome in parallel after mappers find and organize its features. An illustration of the Cloudbreak algorithm working on a simple example is shown in Supplementary Figure **??**.

Cloudbreak can be executed on any Hadoop cluster; Hadoop abstracts away the details of cluster configuration, making distributed applications portable. In addition, our Cloudbreak implementation can leverage the Apache Whirr library to automatically create clusters with cloud service providers such as the Amazon Elastic Compute Cloud (EC2). This enables on demand provisioning of Hadoop clusters which can then be terminated when processing is complete, eliminating the need to invest in a standing cluster and allowing a model in which users can scale their computational infrastructure as their need for it varies over time. Instructions and examples describing how to leverage cloud computing with Cloudbreak are available in the user manual and Supplementary Materials.

2.2 Cloudbreak Implementation

Cloudbreak is a native Java Hadoop application. We deployed Cloudbreak on a 56-node cluster running the Cloudera CDH3 Hadoop distribution, version 0.20.2-cdh3u4. We use snappy compression for MapReduce data. Hadoop's distributed cache mechanism shares the executable files and indices needed for mapping tasks to the nodes in the cluster. To execute the other tools in parallel mode we wrote simple scripts to submit jobs to the cluster using the HTCondor scheduling engine (`http://research.cs.wisc.edu/htcondor/`) with directed acyclic graphs to describe dependencies between jobs.

2.3 Cloudbreak Genotyping

We used the parameters of the fit GMM to infer the genotype of each predicted variant. Assuming that our pipeline is capturing all relevant read mappings near the locus of the variant, the genotype should be indicated by the estimated parameter α, the mixing parameter that controls the weight of the two components in the GMM. We setting a simple cutoff of .35 on the average value of α for each prediction to call the predicted variant homozygous or heterozygous. We used the same cutoff for deletion and insertion predictions.

2.4 Read Simulation

Since there are relatively few heterozygous insertions and deletions annotated in the Venter genome, we used the set of homozygous indels contained in the HuRef data (`HuRef.homozygous_indels.061109.gff`) and randomly assigned each variant to be either homozygous or heterozygous. Based on this genotype, we applied each variant to one or both of two copies of the human GRCh36 chromosome 2 reference sequence. We then simulated paired Illumina reads from these modified references using *dwgsim* from the DNAA software package (`http://sourceforge.net/apps/mediawiki/dnaa/`). We simulated 100bp reads with a mean fragment size of 300bp and a standard deviation of 30bp, and generated 15X coverage for each modified sequence. Pooling the reads from both simulations gives 30X coverage for a diploid sample with a mix of homozygous and heterozygous insertions and deletions.

2.5 Read Alignments

Simulated reads were aligned to hg18 chromosome 2, and NA18507 reads were aligned to the hg19 assembly. Alignments for all programs, unless otherwise noted, were aligned using BWA (Li and Durbin, 2009) `aln` version 0.6.2-r126, with parameter `-e 5` to allow for longer gaps in alignments due to the number of small indels near the ends of larger indels in the Venter data set. GASVPro also accepts ambiguous mappings; we extracted read pairs that did not align concordantly with BWA and re-aligned them with Novoalign V2.08.01, with parameters `-a -r -Ex 1100 -t 250`.

2.6 SV Tool Execution

We ran BreakDancer version 1.1_2011_02_21 in single threaded mode by first executing `bam2cfg.pl` and then running `breakdancer_max` with the default parameter values. To run Breakdancer in parallel mode we first ran `bam2cfg.pl` and then launched parallel instances of `breakdancer_max` for each chromosome using the `-o` parameter. We ran DELLY version 0.0.9 with the `-p` parameter and default values for other parameters. For the parallel run of DELLY we first split the original BAM file with BamTools (Barnett *et al.*, 2011), and then ran instances of DELLY in parallel for each BAM file. We ran GASVPro version 1.2 using the `GASVPro.sh` script and default parameters. Pindel 0.2.4t was executed with default parameters in single CPU mode, and executed in parallel mode for each chromosome using the `-c` option. We executed MoDIL with default parameters except for a `MAX_DEL_SIZE` of 25000, and processed it in parallel on our cluster with a step size of 121475.

2.7 SV Prediction Evaluation

We use the following criteria to define a true prediction given a gold standard set of deletion and insertion variants to test against: A predicted deletion is counted as a true positive if a) it overlaps with a deletion from the gold standard set, b) the length of the predicted call is within 300bp (the library fragment size in both our real and simulated libraries) of the length of the true deletion, and c) the true deletion has not been already been discovered by another prediction from the same method. For evaluating insertions, each algorithm produces insertion predictions that define an interval in which the insertion is predicted to have occurred with start and end coordinates s and e as well as the predicted length of the insertion, l. The true insertions are defined in terms of their actual insertion coordinate i and their actual length l_a. Given this information, we modify the overlap criteria in a) to include overlaps of the intervals $\langle s, \max(e, s + l)\rangle$ and $\langle i, i + l_a\rangle$. In this study we are interested in detecting events larger than 40bp, because with longer reads, smaller events can be more easily discovered by examining gaps in individual reads. Both Pindel and MoDIL make many calls with a predicted event size of under 40bp, so we remove those calls from the output sets of those programs. Finally, we exclude from consideration calls from all approaches that match a true deletion of less than 40bp where the predicted variant length is less than or equal to 75bp in length.

2.8 Preparation of AML Sample

Peripheral blood was collected from a patient with acute myelomonocytic leukemia (previously designated acute myeloid leukemia FAB M4) under a written and oral informed consent process reviewed and approved by the Institutional Review Board of Oregon Health & Science University. Known cytogenetic abnormalities associated with this specimen included trisomy 8 and internal tandem duplications within the FLT3 gene. Mononuclear cells were separated on a Ficoll gradient, followed by red cell lysis. Mononuclear cells were immunostained using antibodies specific for CD3, CD14, CD34, and CD117 (all from BD Biosciences) and cell fractions were sorted using a BD FACSAria flow cytometer. Cell fractions isolated included T-cells (CD3-positive), malignant monocytes (CD14-positive), and malignant blasts (CD34, CD117-positive). We sequenced CD14+ cells on an Illumina Genome Analyzer II, producing 128,819,200 paired-end reads.

2.9 Validation of AML Deletions by PCR

Deletions identified in the AML dataset were validated by PCR. SVs to validate were selected based on calls made by BreakDancer, relying on the score and the number of the reads supporting each single event. Appropriate primers were designed with an internet-based interface, Primer3 (`http://frodo.wi.mit.edu/`), considering the chromosome localization and orientation of the interval involved in the candidate rearrangement. The primers were checked for specificity using the BLAT tool of the UCSC Human Genome Browser (`http://genome.ucsc.edu/cgi-bin/hgBlat`). All the primer pairs were preliminarily tested on the patient genomic DNA and a normal genomic DNA as control. The PCR conditions were as follows: 2 min at 95°C followed by 35 cycles of 30 sec at 95°C, 20 sec at 60°C, and 2 min at 72°C. All the obtained PCR products were sequenced and analyzed by BLAT for sequence specificity.

3 RESULTS

3.1 Tests with Simulated Data

We compared the performance of Cloudbreak for detecting deletions and insertions to a selection of popular tools: the RP method BreakDancer (Chen *et al.*, 2009), GASVPro, an RP method that integrates RD signals and ambiguous mappings (Sindi *et al.*, 2012), the SR method Pindel (Ye *et al.*, 2009), and the hybrid RP-SR method DELLY (Rausch *et al.*, 2012). DELLY produces two sets of calls, one based solely on RP signals, and the other based on RP calls that could be supported by SR evidence; we refer to these sets of calls as DELLY-RP and DELLY-SR. We also attempted to evaluate MoDIL on the same data. All of these methods detect deletions. Insertions can be detected by BreakDancer, Pindel, and MoDIL. See Methods for details on how reads were aligned and each program was invoked. For all alignments we used BWA (Li and Durbin, 2009), although in testing Cloudbreak we have found that the choice of aligner, number of possible mapping locations reported, and whether the reads were aligned in paired-end or single-ended mode can have a variety of effects on the output of the algorithm (Supplementary Fig. **??**).

There is no available test set of real Illumina sequencing data from a sample that has a complete annotation of SVs. Therefore, testing with simulated data is important to fully characterize an algorithm's performance characteristics. On the other hand, any simulated data should contain realistic SVs that follow patterns observed in real data. Therefore, we took one of the most complete lists of SVs from an individual, the list of homozygous insertions and deletions from the genome of J. Craig Venter (Levy *et al.*, 2007), and used it to simulate a 30X read coverage data set for a diploid human Chromosome 2 with a mix of homozygous and heterozygous variants, with 100bp reads and a mean fragment size of 300bp (See Methods).

Figure 2 shows Receiver Operating Characteristic (ROC) curves of the performance of each algorithm at identifying variants larger than 40bp on the simulated data set, as well as the runtimes of the approaches tested, excluding alignment. See Methods for a description of how we identified correct predictions. All approaches show excellent specificity at high thresholds in this simulation. Cloudbreak provides the greatest specificity for deletions at higher levels of sensitivity, followed by DELLY. For insertions, Cloudbreak's performance is similar to or slightly better than Pindel. Cloudbreak's runtime is half that of BreakDancer, the next fastest tool, processing the simulated data in under six minutes. (Of course, Cloudbreak uses many more CPUs as distributed algorithm. See Supplementary Material and Supplementary Table **??** for a discussion of runtimes and parallelization.) The output which we obtained from MoDIL did not have a threshold that could be varied to correlate with the trade-off between precision and recall and therefore it is not included in ROC curves; in addition, MoDIL ran for 52,547 seconds using 250 CPUs in our cluster. Apart from the alignment phase, which is embarrassingly parallel, the feature generation job is the most computationally intensive part of the Cloudbreak workflow. To test its scalability we measured its runtime on Hadoop clusters made up of varying numbers of nodes and observed that linear speedups can be achieved in this portion of the algorithm by adding additional nodes to the cluster until a point of diminishing returns is reached (Supplementary Fig. **??**).

Choosing the correct threshold to set on the output of an SV calling algorithm can be difficult. The use of simulated data and ROC curves allows for some investigation of the performance characteristics of algorithms at varying thresholds. First, we characterized the predictions made by each algorithm at the threshold that gives them maximum sensitivity. For Cloudbreak we chose an operating point at which marginal improvements in sensitivity became very low. The results are summarized in Table 1. MoDIL and Cloudbreak exhibited the greatest recall for deletions. Cloudbreak has also has high precision at this threshold, and discovers many small variants. For insertions, Cloudbreak has the highest recall, although recall is low for all four approaches. Cloudbreak again identifies many small variants. Pindel is the only tool which can consistently identify large insertions, as insertions larger than the library insert size do not produce mapping signatures detectable by RP mapping. We also used the ROC curves to characterize algorithm performance when a low false discovery rate is required. Supplementary Table **??** shows the total number of deletions found by each tool when choosing a threshold that gives an FDR closest to 10% based on the ROC curve. At this more stringent cutoff, Cloudbreak identifies more deletions in every size category than any other tool. Insertions performance never reached an FDR of 10% for any threshold, so insertion predictions are not included in this table. We also examined Cloudbreak's ability to detect events in repetitive regions of the genome, and found that it was similar to the other methods tested (Supplementary Tables **??** and **??**).

It should be noted that the methods tested here vary in their breakpoint resolution (Supplementary Fig. **??**): SR methods have higher resolution than RP methods. Cloudbreak sacrifices additional resolution by dividing the genome into 25bp windows; we believe, however, that increasing sensitivity and specificity is of greatest utility, especially given the emergence of pipelines in which RP calls are validated *in silico* by local assembly.

3.2 Tests with Biological Data

We downloaded a set of reads from Yoruban individual NA18507, experiment ERX009609, from the Sequence Read Archive. This sample was sequenced by Illumina Inc. on the Genome Analyzer II platform with 100bp paired end reads and a mean fragment size (minus adapters) of 300bp, with a standard deviation of 15bp, to a depth of approximately 37X coverage. To create a gold standard set of insertions and deletions to test against, we pooled annotated

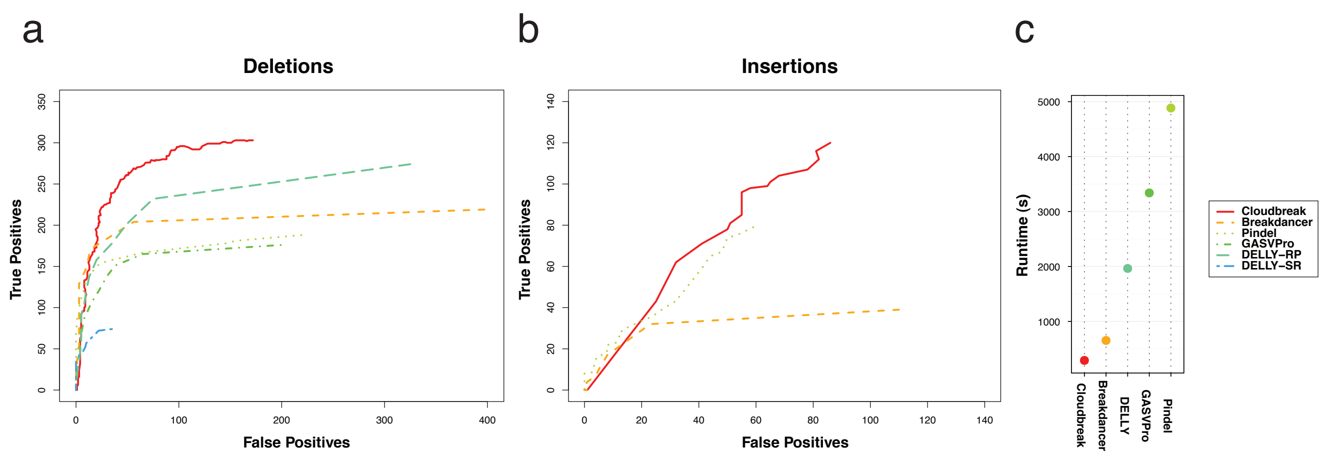

Fig. 2. Accuracy and runtime performance on a simulated data set. (a) Receiver Operating Characteristic (ROC) curves showing the specificity and sensitivity of each tool to deletions larger than 40bp on a simulated set of reads giving diploid coverage of 30X on human chromosome 2. Deletions from the Venter genome were randomly added to one or both haplotypes. Each point on a curve represents a different threshold on the confidence of predictions made by that tool. Thresholds vary by: Cloudbreak - likelihood ratio; BreakDancer, DELLY, GASVPro - number of supporting read pairs; Pindel - simple score. (b) ROC curves for insertion predictions. (c) Runtimes for each tool, not including alignment time, parallelized when possible (see Supplementary Material).

		Prec.	Recall	40-100bp	101-250bp	251-500bp	501-1000bp	$>$ 1000bp
	Total Number			224	84	82	31	26
Deletions	Cloudbreak	0.638	**0.678**	**153** (9)	61 (0)	62 (0)	12 (0)	15 (0)
	BreakDancer	0.356	0.49	89 (0)	54 (0)	53 (0)	8 (0)	15 (0)
	GASVPro	0.146	0.432	83 (2)	32 (0)	55 (0)	8 (0)	15 (0)
	DELLY-RP	0.457	0.613	114 (3)	**68** (0)	**66** (0)	9 (1)	17 (0)
	DELLY-SR	**0.679**	0.166	0 (0)	3 (0)	49 (0)	6 (0)	16 (0)
	Pindel	0.462	0.421	96 (**11**)	24 (0)	48 (0)	5 (0)	15 (0)
	MoDIL	0.132	0.66	123 (6)	66 (**3**)	**66** (**11**)	**17** (**7**)	**23** (**8**)
	Total Number			199	83	79	21	21
Insertions	Cloudbreak	0.451	**0.305**	**79** (**32**)	**32** (**18**)	**11** (8)	1 (0)	0 (0)
	BreakDancer	0.262	0.0968	23 (5)	14 (5)	2 (1)	0 (0)	0 (0)
	Pindel	**0.572**	0.196	52 (25)	5 (1)	10 (**9**)	**3** (**2**)	**9** (**9**)
	MoDIL	0.186	0.0521	14 (1)	4 (0)	1 (0)	2 (**2**)	0 (0)

Table 1. The number of simulated deletions and insertions in the 30X diploid chromosome 2 with Venter indels found by each tool at maximum sensitivity, as well as the number of those variants that were discovered exclusively by each tool (in parentheses). The total number of variants in each size class in the true set of deletions and insertions is shown in the first row of each section.

variants discovered by three previous studies on the same individual. These included data from the Human Genome Structural Variation Project reported by Kidd *et al.* (2008), a survey of small indels conducted by Mills *et al.* (2011b), and insertions and deletions from the merged call set of the phase 1 release of the 1000 Genomes Project (1000 Genomes Project Consortium *et al.*, 2012) which were genotyped as present in NA18507. We merged any overlapping calls of the same type into the region spanned by their unions. We were unable to run MoDIL on the whole-genome data set due to the estimated runtime and storage requirements.

Figure 3 shows the performance of each algorithm at detecting events larger than 40bp on the NA18507 data set. All algorithms show far less specificity for the gold standard set than they did in the single chromosome simulation, although it is difficult to tell how much of the difference is due to the added complexity of real data and a whole genome, and how much is due to missing variants in the gold standard set that are actually present in the sample. For deletions, Cloudbreak is the best performer at the most stringent thresholds, and has the highest or second highest precision at higher sensitivity levels. Cloudbreak has comparable accuracy for insertions to other tools, and can identify the most variants at higher

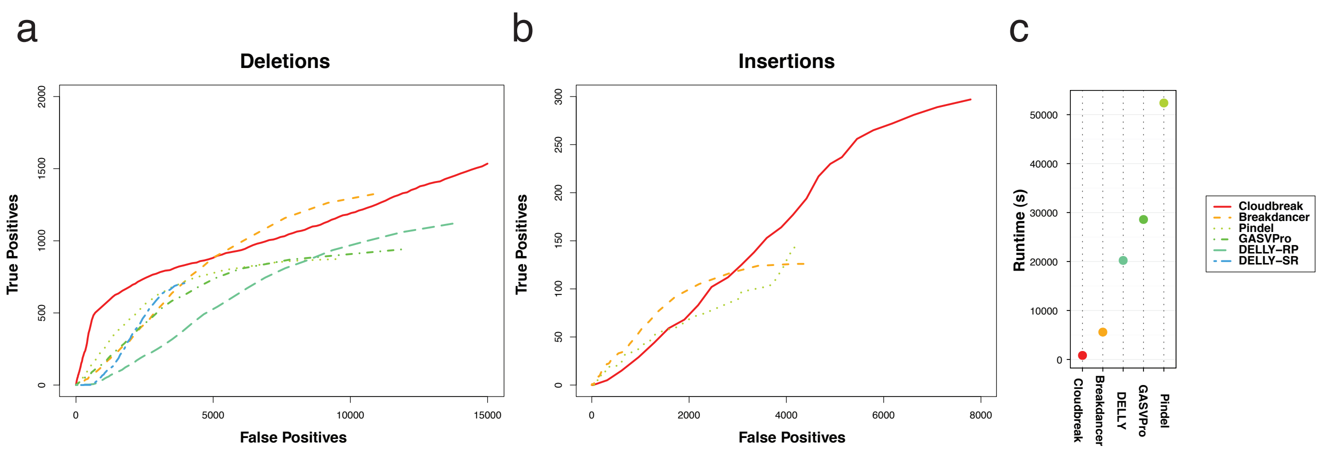

Fig. 3. Accuracy and performance on the 37X NA18507 sample. (a) ROC curve for deletion prediction performance, tested against the combined gold standard sets of deletions taken from Kidd *et al.* (2008), Mills *et al.* (2011b), and 1000 Genomes Project Consortium *et al.* (2012). (b) ROC curve for insertion prediction performance. (c) Runtimes for each tool, not including alignment time, parallelized when possible (see Supplementary Material).

levels of sensitivity. Cloudbreak processes the sample in under 15 minutes on our cluster, more than six times as fast as the next fastest program, BreakDancer, even when BreakDancer is run in parallel for each chromosome on different nodes in the cluster (see Methods and Supplementary Material).

Given the high number of novel predictions made by all tools at maximum sensitivity, we decided to characterize performance at more stringent thresholds. We examined the deletion predictions made by each algorithm using the same cutoffs that yielded a 10% FDR on the simulated chromosome 2 data set, adjusted proportionally for the difference in coverage from 30X to 37X. For insertions, we used the maximum sensitivity thresholds for each tool due to the high observed FDRs in the simulated data. Precision and recall at these thresholds, as well as the performance of each algorithm at predicting variants of each size class, is shown in Table 2. For deletions, Cloudbreak has the greatest sensitivity of any tool, identifying the most variants in each size class. Pindel exhibits the highest precision with respect to the gold standard set. For insertions, Pindel again has the highest precision at maximum sensitivity, while Cloudbreak has by far the highest recall.

3.3 Performance on a Low-Coverage Cancer Data Set

We also tested Cloudbreak on a sequencing data set obtained from a patient with acute myeloid leukemia (AML). This data set consisted of 76bp paired end reads with a mean insert size of 285bp and standard deviation of 50bp, yielding sequence coverage of 5X and physical coverage of 8X. Using a pipeline consisting of Novoalign, BreakDancer, and a set of custom scripts for filtering and annotating candidate SVs, we had previously identified a set of variants present in this sample and validated several using PCR, including 8 deletions. Cloudbreak was able to identify all 8 of the validated deletions, showing that it is still sensitive to variants even when using lower coverage data sets with a greater variance of insert sizes. The variants identified include deletions in the gene CTDSPL/RBPS3, an AML tumor suppressor (Zheng *et al.*, 2012), and NBEAL1, a gene up-regulated in some cancers (Chen *et al.*, 2004). We are currently investigating these deletions to determine their functional impact on this patient.

3.4 Genotyping Variants

Because Cloudbreak explicitly models zygosity in its feature generation algorithm, it can predict the genotypes of identified variants. We tested this on both the simulated and NA18507 data sets. For the NA18507 data set, we considered the deletions from the 1000 Genomes Project, which had been genotyped using the population-scale SV detection algorithm Genome STRiP (Handsaker *et al.*, 2011). Cloudbreak was able to achieve 92.7% and 95.9% accuracy in predicting the genotype of the deletions it detected at our 10% FDR threshold in the simulated and real data sets, respectively. Supplementary Table **??** shows confusion matrices for the two samples using this classifier. None of the three input sets that made up the gold standard for NA18507 contained a sufficient number of insertions that met our size threshold and also had genotyping information. Of the 123 insertions detected by Cloudbreak on the simulated data set, 43 were heterozygous. Cloudbreak correctly classified 78 of the 80 homozygous insertions and 31 of the 43 heterozygous insertions, for an overall accuracy of 88.6%.

4 DISCUSSION

Over the next few years, due to advances in sequencing technology, genomics data are expected to increase in volume by several orders of magnitude, expanding into the realm referred to as “big data”. In addition, the usage of existing information will increase drastically as research in genomics grows and translational applications are developed, and the data sets are reprocessed and integrated into new pipelines. In order to capitalize on this emerging wealth of genome data, novel computational solutions that are capable of scaling with the increasing number and size of these data sets will have to be developed.

		Prec.	Recall	40-100bp	101-250bp	251-500bp	501-1000bp	> 1000bp
	Total Number			7,462	240	232	147	540
Deletions	Cloudbreak	0.0943	**0.17**	**573** (**277**)	**176** (**30**)	**197** (**18**)	**121** (**6**)	**399** (**24**)
	BreakDancer	0.137	0.123	261 (29)	136 (3)	178 (0)	114 (0)	371 (0)
	GASVPro	0.147	0.0474	120 (21)	40 (2)	85 (0)	36 (0)	128 (0)
	DELLY-RP	0.0931	0.1	143 (6)	128 (3)	167 (1)	103 (0)	323 (1)
	DELLY-SR	0.153	0.0485	0 (0)	26 (0)	123 (0)	66 (0)	203 (0)
	Pindel	**0.179**	0.0748	149 (8)	61 (0)	149 (0)	69 (1)	217 (0)
Insertions	Total Number			536	114	45	1	0
	Cloudbreak	0.0323	**0.455**	**265** (**104**)	**49** (**24**)	3 (1)	0 (0)	0 (0)
	BreakDancer	0.0281	0.181	97 (10)	27 (5)	2 (1)	0 (0)	0 (0)
	Pindel	**0.0387**	0.239	144 (45)	14 (7)	**7** (**6**)	**1** (**1**)	0 (0)

Table 2. The precision and recall with respect to the gold standard set of deletions and insertions for each tool on the NA18507 data, as well as the number of variants found in each size class found. Exclusive predictions are in parentheses. For deletions, the same cutoffs were used as for the simulated data as in Supplementary Table **??**, adjusted for the difference in coverage from 30X to 37X. For insertions, the maximum sensitivity cutoff was used.

Among big data infrastructures, MapReduce is emerging as a standard framework for distributing computation across compute clusters. In this paper, we introduced a novel conceptual framework for SV detection algorithms in MapReduce, based on computing local genomic features. This framework provides a scalable basis for developing SV detection algorithms, as demonstrated by our development of an algorithm for detecting insertions and deletions based on fitting a GMM to the distribution of mapped insert sizes spanning each genomic location.

On simulated and real data sets, our approach exhibits high accuracy when compared to popular SV detection algorithms that run on traditional clusters and servers. Detection of insertions and deletions is an important area of research; Mills *et al.* (2011b) recently identified over 220 coding deletions in a survey of a large number of individuals, and they note that such variants are likely to cause phenotypic variation in humans. More SV annotations are also required to assess the impact of SVs on non-coding elements (Mu *et al.*, 2011).

In addition to delivering state-of-the-art performance in a RP-based SV detection tool, our approach offers a basis for developing a variety of SV algorithms that are capable of running in a MapReduce pipeline with the power to process vast amounts of data in a cloud or commodity server setting. With the advent of cloud service providers such as Amazon EC2, it is becoming easy to instantiate on-demand Hadoop compute clusters. Having computational approaches that can harness this capability will become increasingly important for researchers or clinicians who need to analyze increasing amounts of sequencing data.

5 CONCLUSION

ACKNOWLEDGEMENT

Text Text Text Text Text Text Text Text. **?** might want to know about text text text text

Funding: Text Text Text Text Text Text Text Text.

REFERENCES

1000 Genomes Project Consortium, Abecasis, G. R., Auton, A., Brooks, L. D., DePristo, M. A., Durbin, R. M., Handsaker, R. E., Kang, H. M., Marth, G. T., and McVean, G. A. (2012). An integrated map of genetic variation from 1,092 human genomes. *Nature*, **491**(7422), 56–65.

Abyzov, A., Urban, A. E., Snyder, M., and Gerstein, M. (2011). CNVnator: an approach to discover, genotype, and characterize typical and atypical CNVs from family and population genome sequencing. *Genome Res.*, **21**(6), 974–984.

Alkan, C., Kidd, J. M., Marques-Bonet, T., Aksay, G., Antonacci, F., Hormozdiari, F., Kitzman, J. O., Baker, C., Malig, M., Mutlu, O., Sahinalp, S. C., Gibbs, R. A., and Eichler, E. E. (2009). Personalized copy number and segmental duplication maps using next-generation sequencing. *Nat. Genet.*, **41**(10), 1061–1067.

Alkan, C., Coe, B. P., and Eichler, E. E. (2011). Genome structural variation discovery and genotyping. *Nat. Rev. Genet.*, **12**(5), 363–376.

Barnett, D. W., Garrison, E. K., Quinlan, A. R., Stromberg, M. P., and Marth, G. T. (2011). BamTools: a C++ API and toolkit for analyzing and managing BAM files. *Bioinformatics*, **27**(12), 1691–1692.

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

Chen, J., Lu, Y., Xu, J., Huang, Y., Cheng, H., Hu, G., Luo, C., Lou, M., Cao, G., Xie, Y., and Ying, K. (2004). Identification and characterization of NBEAL1, a novel human neurobeachin-like 1 protein gene from fetal brain, which is up regulated in glioma. *Mol Brain Res*, **125**(1-2), 147–155.

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., Shi, X., Fulton, R. S., Ley, T. J., Wilson, R. K., Ding, L., and Mardis, E. R. (2009). BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nat. Methods*, **6**(9), 677–681.

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

Chiara, M., Pesole, G., and Horner, D. S. (2012). SVM^2: an improved paired-end-based tool for the detection of small genomic structural variations using high-throughput single-genome resequencing data. *Nucleic Acids Res.*, **40**(18), e145.

Conrad, D. F., Pinto, D., Redon, R., Feuk, L., Gokcumen, O., Zhang, Y., Aerts, J., Andrews, T. D., Barnes, C., Campbell, P., Fitzgerald, T., Hu, M., Ihm, C. H., Kristiansson, K., MacArthur, D. G., MacDonald, J. R., Onyiah, I., Pang, A. W. C., Robson, S., Stirrups, K., Valsesia, A., Walter, K., Wei, J., Wellcome Trust Case

Control Consortium, Tyler-Smith, C., Carter, N. P., Lee, C., Scherer, S. W., and Hurles, M. E. (2010). Origins and functional impact of copy number variation in the human genome. *Nature*, **464**(7289), 704–712.

Dean, J. and Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. *Commun. ACM*, **51**(1), 107–113.

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

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.

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

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

Langmead, B., Schatz, M. C., Lin, J., Pop, M., and Salzberg, S. L. (2009). Searching for SNPs with cloud computing. *Genome Biol.*, **10**(11), R134.

Langmead, B., Hansen, K. D., and Leek, J. T. (2010). Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biol.*, **11**(8), R83.

Lee, S., Hormozdiari, F., Alkan, C., and Brudno, M. (2009). MoDIL: detecting small indels from clone-end sequencing with mixtures of distributions. *Nat. Methods*, **6**(7), 473–474.

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

Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, **25**(14), 1754–1760.

Marschall, T., Costa, I. G., Canzar, S., Bauer, M., Klau, G. W., Schliep, A., and Schönhuth, A. (2012). CLEVER: clique-enumerating variant finder. *Bioinformatics*, **28**(22), 2875–2882.

McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., and DePristo, M. A. (2010). The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.*, **20**(9), 1297–1303.

Michaelson, J. J. and Sebat, J. (2012). forestSV: structural variant discovery through statistical learning. *Nat. Methods*, **9**(8), 819–821.

Mills, R. E., Walter, K., Stewart, C., Handsaker, R. E., Chen, K., Alkan, C., Abyzov, A., Yoon, S. C., Ye, K., Cheetham, R. K., Chinwalla, A., Conrad, D. F., Fu, Y., Grubert, F., Hajirasouliha, I., Hormozdiari, F., Iakoucheva, L. M., Iqbal, Z., Kang, S., Kidd, J. M., Konkel, M. K., Korn, J., Khurana, E., Kural, D., Lam, H. Y. K., Leng, J., Li, R., Li, Y., Lin, C.-Y., Luo, R., Mu, X. J., Nemesh, J., Peckham, H. E., Rausch, T., Scally, A., Shi, X., Stromberg, M. P., Stütz, A. M., Urban, A. E., Walker, J. A., Wu, J., Zhang, Y., Zhang, Z. D., Batzer, M. A., Ding, L., Marth, G. T., McVean, G., Sebat, J., Snyder, M., Wang, J., Ye, K., Eichler, E. E., Gerstein, M. B., Hurles, M. E., Lee, C., McCarroll, S. A., and Korbel, J. O. (2011a). Mapping copy number variation by population-scale genome sequencing. *Nature*, **470**(7332), 59–65.

Mills, R. E., Pittard, W. S., Mullaney, J. M., Farooq, U., Creasy, T. H., Mahurkar, A. A., Kemeza, D. M., Strassler, D. S., Ponting, C. P., Webber, C., and Devine, S. E. (2011b). Natural genetic variation caused by small insertions and deletions in the human genome. *Genome Res.*, **21**(6), 830–839.

Mu, X. J., Lu, Z. J., Kong, Y., Lam, H. Y. K., and Gerstein, M. B. (2011). Analysis of genomic variation in non-coding elements using population-scale sequencing data from the 1000 Genomes Project. *Nucleic Acids Research*, **39**(16), 7058–7076.

Quinlan, A. R., Clark, R. A., Sokolova, S., Leibowitz, M. L., Zhang, Y., Hurles, M. E., Mell, J. C., and Hall, I. M. (2010). Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome. *Genome Res.*, **20**(5), 623–635.

Rausch, T., Zichner, T., Schlattl, A., Stütz, A. M., Benes, V., and Korbel, J. O. (2012). DELLY: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, **28**(18), i333–i339.

Schatz, M. C. (2009). CloudBurst: highly sensitive read mapping with MapReduce. *Bioinformatics*, **25**(11), 1363–1369.

Sindi, S., Helman, E., Bashir, A., and Raphael, B. J. (2009). A geometric approach for classification and comparison of structural variants. *Bioinformatics*, **25**(12), i222–30.

Sindi, S. S., Onal, S., Peng, L., Wu, H.-T., and Raphael, B. J. (2012). An integrative probabilistic model for identification of structural variation in sequencing data. *Genome Biol.*, **13**(3), R22.

Wang, J., Mullighan, C. G., Easton, J., Roberts, S., Heatley, S. L., Ma, J., Rusch, M. C., Chen, K., Harris, C. C., Ding, L., Holmfeldt, L., Payne-Turner, D., Fan, X., Wei, L., Zhao, D., Obenauer, J. C., Naeve, C., Mardis, E. R., Wilson, R. K., Downing, J. R., and Zhang, J. (2011). CREST maps somatic structural variation in cancer genomes with base-pair resolution. *Nat. Methods*, **8**(8), 652–654.

Ye, K., Schulz, M. H., Long, Q., Apweiler, R., and Ning, Z. (2009). Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads. *Bioinformatics*, **25**(21), 2865–2871.

Yoon, S., Xuan, Z., Makarov, V., Ye, K., and Sebat, J. (2009). Sensitive and accurate detection of copy number variants using read depth of coverage. *Genome Res.*, **19**(9), 1586–1592.

Zheng, Y.-S., Zhang, H., Zhang, X.-J., Feng, D.-D., Luo, X.-Q., Zeng, C.-W., Lin, K.-Y., Zhou, H., Qu, L.-H., Zhang, P., and Chen, Y.-Q. (2012). MiR-100 regulates cell differentiation and survival by targeting RBSP3, a phosphatase-like tumor suppressor in acute myeloid leukemia. *Oncogene*, **31**(1), 80–92.