BIOINFORMATICS ORIGINAL PAPER

Vol. 30 no. 19 2014, pages 2787–2795
doi:10.1093/bioinformatics/btu345

Data and text mining

Advance Access publication June 3, 2014

SMASH: a benchmarking toolkit for human genome variant calling

Ameet Talwalkar[1,*,†], Jesse Liptrap[1,†], Julie Newcomb[1], Christopher Hartl[1,2], Jonathan Terhorst[3], Kristal Curtis[1], Ma'ayan Bresler[1], Yun S. Song[1,3], Michael I. Jordan[1,3] and David Patterson[1,*]

[1]Department of Electrical Engineering and Computer Science, UC Berkeley, Berkeley, CA 94720, USA, [2]The Broad Institute of Harvard and MIT, Cambridge, MA 02142, USA and [3]Department of Statistics, UC Berkeley, Berkeley, CA 94720, USA

Associate Editor: Michael Brudno

ABSTRACT

Motivation: Computational methods are essential to extract actionable information from raw sequencing data, and to thus fulfill the promise of next-generation sequencing technology. Unfortunately, computational tools developed to call variants from human sequencing data disagree on many of their predictions, and current methods to evaluate accuracy and computational performance are *ad hoc* and incomplete. Agreement on benchmarking variant calling methods would stimulate development of genomic processing tools and facilitate communication among researchers.

Results: We propose SMASH, a benchmarking methodology for evaluating germline variant calling algorithms. We generate synthetic datasets, organize and interpret a wide range of existing benchmarking data for real genomes and propose a set of accuracy and computational performance metrics for evaluating variant calling methods on these benchmarking data. Moreover, we illustrate the utility of SMASH to evaluate the performance of some leading single-nucleotide polymorphism, indel and structural variant calling algorithms.

Availability and implementation: We provide free and open access online to the SMASH tool kit, along with detailed documentation, at smash.cs.berkeley.edu

Contact: ameet@cs.berkeley.edu or pattrsn@cs.berkeley.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on October 25, 2013; revised on April 30, 2014; accepted on May 13, 2014

1 INTRODUCTION

Next-generation sequencing is revolutionizing biological and clinical research. Long hampered by the difficulty and expense of obtaining genomic data, life scientists now face the opposite problem: faster, cheaper technologies are beginning to generate massive amounts of new sequencing data that are overwhelming our technological capacity to conduct genomic analyses (Mardis, 2010). Computational processing will soon become the bottleneck in genome sequencing research, and as a result, computational biologists are actively developing new tools to more efficiently and accurately process human genomes and call variants, e.g. SAMTools (Li *et al.*, 2009), GATK (DePristo *et al.*, 2011), Platypus (http://www.well.ox.ac.uk/platypus), BreakDancer (Chen *et al.*, 2009), Pindel (Ye *et al.*, 2009) and Dindel (Albers *et al.*, 2011).

Unfortunately, single-nucleotide polymorphism (SNP) callers disagree as much as 20% of the time (Lyon *et al.*, 2012), and there is even less consensus in the outputs of structural variant algorithms (Alkan *et al.*, 2011). Moreover, reproducibility, interpretability and ease of setup and use of existing software are pressing issues currently hindering clinical adoption (Nekrutenko and Taylor, 2012). Indeed, reliable benchmarks are required to measure accuracy, computational performance and software robustness, and thereby improve them.

In an ideal world, benchmarking data to evaluate variant calling algorithms would consist of several fully sequenced, perfectly known human genomes. However, ideal validation data do not exist in practice. Technical limitations, such as the difficulty in accurately sequencing low-complexity regions, along with budget constraints, such as the cost to generate high-coverage Sanger reads, limit the quality and scope of validation data. Nonetheless, significant resources have already been devoted to generate subsets of benchmarking data that are substantial enough to drive algorithmic innovation. Alas, the existing data are not curated, thus making it extremely difficult to access, interpret and ultimately use for benchmarking purposes.

Owing to the lack of curated ground truth data, current benchmarking efforts with sequenced human genomes are lacking. The majority of benchmarking today relies on either simulated data or a limited set of validation data associated with real-world datasets. Simulated data are valuable but do not tell the full story, as variant calling is often substantially easier using synthetic reads generated via simple generative models. Sampled data, as mentioned earlier, are not well curated, resulting in benchmarking efforts, such as the Genome in a Bottle Consortium (Zook and Salit, 2011) and the Comparison and Analytic Testing resource (GCAT) (http://www.bioplanet.com/gcat), that rely on a single dataset with a limited quantity of validation data.

Rigorously evaluating predictions against a validation dataset presents several additional challenges. Consensus-based evaluation approaches, used in various benchmarking efforts (The 1000 Genomes Project Consortium, 2010; DePristo *et al.*, 2011; Kedes and Campany, 2011), may be misleading. Indeed,

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

© The Author 2014. Published by Oxford University Press. All rights reserved. For Permissions, please e-mail: journals.permissions@oup.com

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

different methods may in fact make similar errors, a fact that remains hidden without ground truth data. In cases where 'noisy' ground truth data are used, e.g. calls based on Sanger sequencing with some known error rate or using SNP chips with known error rates, accuracy metrics should account for the effect of this noise on predictive accuracy. Additionally, given the inherent ambiguity in the Variant Calling Format (VCF) format used to represent variants, evaluation can be quite sensitive to the (potentially inconsistent) representations of predicted and ground truth variants. Moreover, owing to the growing need to efficiently process raw sequencing data, computational performance is an increasingly important yet to date largely overlooked factor in benchmarking. There currently exist no benchmarking methodologies that—in a consistent and principled fashion—account for noise in validation data, ambiguity in variant representation or computational efficiency of variant calling methods.

Without any standard datasets and evaluation methodologies, research groups inevitably perform *ad hoc* benchmarking studies, working with different datasets and accuracy metrics, and performing studies on a variety of computational infrastructures. Competition-based exercises (Earl *et al*., 2011; Kedes and Campany, 2011) are a popular route for benchmarking that aim to address some of these inconsistencies, but they are ephemeral by design and often suffer from the same data and evaluation pitfalls described earlier.

In short, the lack of consistency in datasets, computational frameworks and evaluation metrics across the field prevents simple comparisons across methodologies, and in this work, we make a first attempt at addressing these issues. We propose SMASH, a standard methodology for benchmarking variant calling algorithms based on a suite of **S**ynthetic, **M**ouse **a**nd **S**ampled **H**uman data. SMASH leverages a rich set of validation resources, in part bootstrapped from the patchwork of existing data. We provide free and open access to SMASH, which consists of:

- A set of five full genomes with associated deep coverage short-read datasets (real and synthetic);
- Three contaminated variants of these datasets that mimic real-world use cases (M.DePristo, 2013, personal communication) and test the robustness of variant callers in terms of accuracy and required computational resources;
- Ground truth validation data for each genome along with detailed error profiles;
- Accuracy metrics that account for the uncertainty in validation data;
- Methodology to resolve the ambiguity in variant representations, resulting in stable measurements of accuracy; and
- Performance metrics to measure computational efficiency (and implicitly measure software robustness) that leverage the Amazon Web Services (AWS) cloud computing environment.

SMASH is designed to facilitate progress in algorithm development by making it easier for researchers to evaluate their systems against each other.

2 METHODS

2.1 Benchmarking datasets

In this section, we describe the benchmarking datasets contained within SMASH. A 'benchmarking dataset' consists of three components. The first two components are the inputs to the variant calling algorithm to be benchmarked, namely, short reads generated from next-generation sequencing technology and a reference genome (used for alignment and variant representation). The third component is the validation data (represented via the standard VCF format) that are used to evaluate the quality of an algorithm's predictions.

The left panel of Figure 1 illustrates the three desired properties of a benchmarking dataset. Ideally, we would like to evaluate variant calling performance on a human genome (**H**), have access to comprehensive validation (**C**) of the underlying sequenced genome and call variants using real reads (**R**), i.e. reads generated by an actual sequencing machine and not a simulator. To the best of our knowledge, no existing dataset satisfies all three properties. Instead, SMASH consists of three types of benchmarking datasets that satisfy two of these three properties, as depicted in the three panels on the right of Figure 1. Additionally, for each type of dataset, we also include a contaminated version in which the short reads are contaminated with reads from a separate genome, mimicking the impurities that can be introduced in practice while preparing a sample and/or using a contaminated sequencing machine, and thus testing the robustness of variant callers in this challenging and realistic setting. Table 1 summarizes our validation data, and we next provide details about these datasets.

2.1.1 Synthetic datasets We derive our synthetic datasets from J. Craig Venter's genome (HuRef). HuRef variants provided by Levy *et al*. (2007) are represented as variations relative to the human reference genome (in VCF format). We create an unphased diploid sample genome by starting with two copies of the hg19 reference genome and inserting each HuRef variant into one or both of these copies depending on its zygosity. We simulate Illumina reads from this diploid sample genome using simNGS (http://www.ebi.ac.uk/goldman-srv/simNGS) with its default settings. Our second dataset uses the same validation data as the first, along with a version of Venter's short-read data contaminated by similar short-read data derived from an approximation of James Watson's genome.

***Validation Error Profile*:** Any errors in HuRef will be carried through to our synthetic genome, and it is likely that errors in HuRef are sequence-context specific. Nonetheless, as the sample genome and the reads are synthetically generated, the VCF files contain noiseless ground truth data.

2.1.2 Mouse datasets The mouse datasets leverage existing mouse genomic data associated with the canonical mouse reference as well as from the Mouse Genomes Project (Yalcin *et al*., 2011). From these data, we create benchmarking datasets with real reads and with comprehensive validation, using the canonical homozygous mouse reference as our

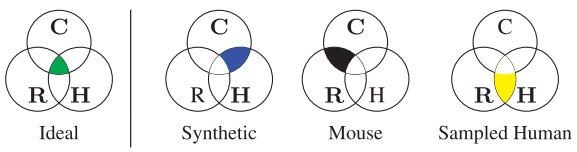

Fig. 1. An 'ideal' benchmarking dataset satisfies three properties: it contains real reads (**R**), it includes comprehensive validation of the underlying genome (**C**) and its underlying genome is human (**H**). SMASH contains three types of benchmarking datasets, each of which satisfies two of the three desirable properties of an ideal dataset, so as to cover all three properties

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

Table 1. Summary of SMASH's validation datasets

Type	Genome	Validation error	Sequencer	Length (bp)	Insert size (bp)	Coverage (×)
Synthetic	Venter	None	SimNGS	101	400	30
	Contam. Venter					
Mouse	B6 strain	0.2% (SNP/Indel), 0.3% (SV)	GAIIx	101	–34	58.6
	Contam. B6 strain					
Human	NA12878	0.04% (SNP), 1% (SV)	HiSeq2000	101	300	50
	Contam. NA12878		HiSeq2000	101	300	50
	NA18507		HiSeq2500	100	300	44
	NA19240		HiSeq2000	101	296	49

sample genome (Church et al., 2009). Our first dataset consists of paired-end reads from the B6 mouse strain (Gnerre *et al*., 2011), a VCF derived from differences between the mouse reference (based on the B6 mouse strain), and a 'fake' reference we created using an alternative mouse strain (the DBA mouse strain). Figure 2 illustrates the process by which we create this dataset, and further details are provided in Supplementary Material B.2. Our second dataset uses the same validation data as the first, along with a version of the B6 short reads contaminated by short reads corresponding to a human genome (NA12878). We use these human reads because, to the best of our knowledge, they are the only publicly available reads generated by the same sequencing methodology as the mouse reads (Gnerre *et al*., 2011).

Validation Error Profile: There are two main sources of error in this dataset, namely, errors in the mouse reference genome itself and genetic differences between the mouse reference genome and the individual from which short reads were produced. Based on calculations detailed in Supplementary Material B.2, we upper bound the error rates for SNPs and indels at 0.2% and the error rate for Structural Variants (SVs) at 0.3%. Finally, it is worth noting that there are systematic differences between mouse and human genomes. Mouse segmental duplication is more intrachromosomal, and human intrachromosomal duplication is more high-identity (Church *et al*., 2009). As a result, variant calling performance may vary between the mouse and human datasets.

2.1.3 Sampled human datasets Our real human genomes consist of three well-studied human genomes, including a European female (NA12878), a Nigerian male (NA18507) and a Nigerian female (NA19240). Short reads for NA12878 and NA18507 were obtained from Illumina's Platinum Genomes (http://www.illumina.com/platinum genomes/); the NA19240 short reads are available from Sequence Read Archive (SRA) (http://www.ncbi.nlm.nih.gov/sra/SRX152746). Our validation data consist of subsets of validated SNP and SV information. SMASH also includes high-coverage Illumina reads for each of these datasets.

We derive our validated SNPs from the intersection of calls from two SNP chips from HapMap2: Perlegen and Illumina BeadArray (Frazer *et al*., 2007). We chose these chips owing to the substantial intersection of their call sites, and because their calls could be readily disambiguated, unlike the two chip technologies used in HapMap3 (HapMap Consortium, 2010). The intersection of their results in NA12878 yields 132K calls, of which 55K are non-reference. Our SV validation data consist of 169 insertions and deletions called from the alignment of finished fosmid sequences (Kidd *et al*., 2010a, b).

Finally, we include a contaminated version of our NA12878 dataset, in which the NA12878 short reads are contaminated by short reads corresponding to this individual's husband (NA12877), generated by the same sequencing methodology by Illumina's Platinum Genomes project.

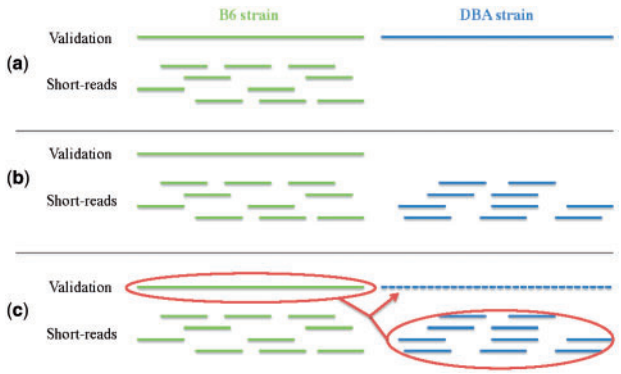

Fig. 2. Schematic illustrating process by which SMASH's first 'Mouse' dataset is generated. (**a**) Our ideal setup in which the B6 strain (with comprehensive validation and corresponding short reads) serves as the sample and the DBA strain serves as the reference. (**b**) Publicly available data (note that the B6 validation data are the canonical mouse reference). (**c**) Construction of an approximate DBA validation set (the 'fake' reference) by leveraging a rough set of variants for the DBA strain called relative to the canonical reference

Validation Error Profile: The error in our validated SNP data is owing to errors in the underlying SNP chip technologies used to generate the data. Moreover, there are various sources of error for our validated SVs, associated with generating and processing the fosmid sequences. As detailed in Supplementary Material B.3, we upper bound the error rate for SNPs at 0.04% and the error rate for SVs at 1.0%.

2.2 Evaluation metrics

We now discuss our set of evaluation metrics of variant calling algorithms against the benchmarking datasets described in Section 2.1. We propose the use of both accuracy and computational performance metrics. Moreover, although SMASH focuses on reference-based variant calling methods, it *does not include alignment-specific metrics*, as alignment is an intermediate step (albeit an important one) in the process of variant calling. We believe that improvements in alignment should be measured as a function of their impact on variant calling, both in terms of accuracy and computational performance.

2.2.1 Accuracy We report two standard metrics from information retrieval. The first metric, *recall*, measures the 'probability of calling a validated variant', while the second metric, *precision*, measures the 'probability that a called variant is correct' (See Section C in the Supplementary Material for a more detailed discussion of the use of recall and precision in the context of variant calling.). For SNPs and

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

indels (50 bp or less), we measure alternate alleles and exact breakpoints, thus checking zygosity but ignoring phasing. For structural variants, we ignore zygosity, and evaluate left breakpoint and length, both approximately and exactly. Specifically, we report results with error tolerances of 0 (exact), 100 and 1000 bp to highlight the accuracy of variant callers at different resolutions. In some situations, such as our human SNP or SV validation data, the validation data have positive labels but little or no negative labels, and in these situations, only recall (and not precision) is reported. Additionally, our computation of recall does not explicitly take into account the impact of sampling in the context of SMaSH's sampled human benchmarking datasets.

2.2.2 Computational performance We use two chief metrics to measure computational performance, namely, *hours per genome* and *dollars per genome*, and we benchmark performance on AWS. AWS' cloud infrastructure allows for reproducible benchmarking and ensures robust implementations of variant calling algorithms. When using SMaSH, researchers can benchmark algorithms on AWS using their preferred compute instances, such as single core, multicore, Graphics processing unit (GPU) or distributed cluster, and AWS' pricing mechanism naturally dictates the tradeoff between cost and time. We additionally report more fine-grained performance metrics to help researchers optimize their AWS configuration, including clock time, CPU time, the maximum number of threads used, the maximum disk space required and the maximum and average amount of memory used during the run of the algorithm.

2.2.3 Accounting for noisy validation data The performance of an algorithm can only be quantified up to the level of noise in the validation data itself. Because we are working with noisy validation data, it is crucial to capture this uncertainty when reporting results. To do so, we assume that we have an estimate for the number E of validation errors. In settings where we only have positive labels, this estimate captures the number of positive labels that are incorrectly genotyped (either a positive label where there should be none, or a validated variant that is correctly located but incorrectly genotyped). In settings where we have access to both positive and negative labels, this estimate measures either omissions or errors of the aforementioned type.

To quantify these errors, we first note that each called variant can be described via two sets of labels: positive or negative, depending solely on the caller, and true or false, depending also on the validation dataset. We can compute true positives (TPs), false positives (FPs) and false negatives (FNs) from these two sets of labels. Proposition 1 presents bounds on recall and precision given E and in terms of TP, FP and FN. These bounds hold generally for SNPs, indels and SVs evaluation, and for various evaluation metrics, e.g. metrics considering zygosity and insertion sequence (see Section D in the Supplementary Material for further details and proof).

PROPOSITION 1. Let present = TP + FN be the number of positive labels, and let P = TP + FP be the number of positive calls. In the case of only positive validated labels,

$$\frac{\mathrm{TP}-E}{\mathrm{present}} \leq \mathrm{recall} \leq \begin{cases} \dfrac{\mathrm{TP}+E}{\mathrm{present}} & \text{if } E \leq \mathrm{FN} \\ \dfrac{\mathrm{TP}}{\mathrm{present}-E} & \text{otherwise} \end{cases}$$

In the case of both positive and negative labels, the same recall bounds apply, and the following precision bounds hold:

$$\frac{\mathrm{TP}-E}{P} \leq \mathrm{precision} \leq \frac{\mathrm{TP}+E}{P} \quad (1)$$

Proposition 1 states that recall and precision have worst-case additive errors of the form E/present and E/P, respectively. We use these bounds when reporting results in Section 3.

2.2.4 Ambiguity resolution SMaSH incorporates three steps in the evaluation process to minimize the impact of VCF ambiguity. The first two steps, cleaning and left normalization, are (standard) VCF preprocessing steps that we perform independently on both the ground truth VCF and the predicted VCF. The cleaning step removes ambiguity associated with case discrepancies, and also filters out extraneous VCF entries, i.e. homozygous reference calls and calls where the reference and alternate alleles match. Left normalization involves left shifting all variants as far as possible and is the VCF standard, although this convention is not followed by all variant calling algorithms. Left normalization removes certain types of ambiguity associated with indels and SVs, such as by unambiguously representing the deletion 'GCGCGC → GCGC' as a deletion event associated with the two leftmost 'GC' bases.

Our final step is a novel ambiguity resolution algorithm, RESCUE, which involves a second pass over the VCF files during evaluation. After initially strictly comparing the calls between the predicted and the true VCFs, we aim to 'rescue' variants marked as incorrect (both FPs and FNs) owing to VCF ambiguity. For each such call, we create two short sequences by expanding the full sequence in some short window around the call in the true and predicted VCF, respectively. We then rescue the call if the two sequences are equivalent. Rescued calls are thus by definition correct, and notably, RESCUE can only improve the quality of the reported precision and recall figures. Nonetheless, the number of calls that RESCUE is able to rescue is dependent on the window size, as discussed in Section 3.1.

See Section E in the Supplementary Material for further algorithmic details, including discussions about edge cases such as overlapping alleles and combinations of alleles that cancel out.

2.3 Usage

All the materials necessary to run SMaSH are available at our Web site, `smash.cs.berkeley.edu`. All relevant files are available for download, including Burrows-Wheeler Aligner (BWA)-aligned BAM files containing the raw reads, ground truth VCF files and reference files. All scripts used to calculate results are available in a public repository at `github.com/amplab/smash`, including evaluation, rescue, VCF normalization and contamination scripts. All data included in SMaSH are derived from publicly available sources and thus can be freely redistributed.

We provide detailed instructions for running SMaSH on AWS. For a researcher wishing to benchmark a new aligner, we describe how to download the short reads, run the aligner on AWS, execute some or all of the variant callers evaluated in Section 3 and run the SMaSH evaluation scripts to get performance and accuracy metrics. In contrast, for a researcher with a new variant caller, we describe how to download our aligned BAM files, run the caller on AWS and evaluate the overall performance and accuracy.

Finally, we plan to update SMaSH as new validated datasets become available. We also invite users to submit performance and accuracy results associated with new aligner and variant caller pipelines to our results page.

3 RESULTS

In this section, we first evaluate the impact of our ambiguity resolution algorithms. We next illustrate the utility of SMaSH by evaluating the performance of some leading SNP, indel and structural variant calling algorithms. The goal of these

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

experiments is to highlight SMASH's functionality, and to simplify the discussion, we use the default settings for all variant calling algorithms and the same machine instance in all of our Amazon Elastic Compute Cloud (EC2) experiments. We note that improved accuracy and computational performance results may indeed be possible via parameter tuning and optimizing to minimize EC2 instance footprints.

In all reported results, evaluation is reported as described in Section 2.2. See Section G in the Supplementary Material for further implementation details.

3.1 Ambiguity resolution

We first examine the precision and recall rates for mpileup and GATK as a function of this user-specified window parameter, as Table 2 illustrates. We see that precision and recall are quite robust to various window sizes. Nonetheless, for small windows ($\leq$ 25 bp), comparatively fewer TPs are rescued, likely owing to the omission of variants associated with an ambiguously represented event. Meanwhile, large windows ($\geq$ 150 bp) also lead to fewer rescues, likely owing to nearby FPs unrelated to the ambiguously represented event entering the window. We observe that a 50–100 bp window balances these two tradeoffs, and we use a 50 bp window in all subsequent experiments for computational reasons.

Next, Table 3 shows the impact of each of the three ambiguity resolution steps. The results show that all three steps significantly impact the precision and recall of both GATK (DePristo *et al.*, 2011) and mpileup (Li *et al.*, 2009) on calling indels for the mouse dataset. Our ambiguity resolution has a similar impact on indel detection for the Venter dataset, and also has a significant (although less drastic) impact on SNP detection for both datasets (See Section E in the Supplementary Material for further results).

3.2 SNP calling

We benchmark the performance of GATK and mpileup to call SNPs, and Table 4 summarizes the results. The results show that GATK is more computationally expensive, but does not strictly outperform mpileup on the uncontaminated datasets. Moreover, the effect of contamination on precision is fairly visible on Venter, where mpileup's accuracy clearly degrades while GATK appears robust to the contamination. The difference in performance on the mouse and contaminated mouse dataset is less pronounced, because, as noted in Section F (Supplementary Material), the aligner was able to filter the contaminated reads before they were processed by mpileup or GATK.

3.3 Indel calling

We evaluate the performance of mpileup, GATK and Pindel on the detection of indels, in particular insertions or deletions of 50 bp or less, with detailed results presented in Table 5 and Supplementary Tables S6 and S7 (SMASH's sampled human benchmarking datasets do not include any validated indels, and so the indel results are restricted to the mouse and synthetic human datasets.). These results demonstrate that GATK and Pindel outperform mpileup in terms of accuracy, but are more expensive computationally. In fact, Pindel highlights the importance of SMASH's computational performance metrics, as it failed to complete within our predetermined time limit of 400 h (i.e. a $1000 AWS budget) on both the contaminated Venter and mouse datasets, and we thus did not obtain results for these experiments. On both contaminated Venter and contaminated mouse, mpileup and GATK gained slightly in precision and worsened in recall as compared with their uncontaminated counterparts. This difference seems to result from the fact that both algorithms predicted fewer indels overall on the contaminated sets than they did SNPs; for example, mpileup called 2.5% fewer indel deletions on mouse and only 0.5% fewer SNPs. Consistent with the SNP results, however, mpileup was less robust to contamination than GATK.

3.4 Structural variant calling

We benchmark Pindel and BreakDancerMax on insertions and deletions >50 bp in length, reporting approximate breakpoint results with tolerance of 100 bp in Table 6 and Supplementary Table S10. We first note that we do not report results for the several experiments that ran longer than our 400 h budget, namely, BreakDancer on contaminated NA12878 and Pindel on mouse, contaminated Venter and contaminated NA12878. On the experiments that did complete, we observe that (perhaps unsurprisingly) SV accuracy is much lower than SNP and indel accuracies, and in particular, both Pindel and BreakDancer have fairly low accuracy on long insertions. Indeed, for NA12827, BreakDancer misses all insertions even though it accurately

Table 2. Effect of the window size parameter in the RESCUE algorithm on indel precision and recall for mpileup and GATK on the Venter genome

Window	mpileup				GATK			
	Insertions		Deletions		Insertions		Deletions	
	Prec (%)	Rec (%)	Prec (%)	Rec (%)	Prec (%)	Rec (%)	Prec (%)	Rec (%)
25	85.8	70.9	91.2	75.2	90.3	87.8	91.9	90.6
50	85.8	70.9	91.3	75.2	90.6	88.1	92.3	90.9
100	85.8	70.9	91.3	75.2	90.6	88.1	92.3	90.9
150	85.7	70.9	91.3	75.2	90.5	88.1	92.2	90.8

Note: Error bounds are excluded, as there is no uncertainty in the Venter validation data.

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

Table 3. Effect of ambiguity resolution on benchmarking GATK and mpileup on indels using the mouse dataset

Strategy	mpileup				GATK			
	Insertions		Deletions		Insertions		Deletions	
	Pre	Rec	Pre	Rec	Pre	Rec	Pre	Rec
Cleaning	81.1 ± 3.5	12.2 ± 0.5	75.2 ± 3.3	12.6 ± 0.6	73.1 ± 0.4	86.4 ± 0.5	68.9 ± 0.4	91.2 ± 0.6
Normalization	76.6 ± 0.5	66.5 ± 0.4	76.6 ± 0.5	74.9 ± 0.5	85.7 ± 0.4	84.0 ± 0.4	80.5 ± 0.4	89.9 ± 0.5
RESCUE	87.8 ± 0.5	76.6 ± 0.4	79.0 ± 0.4	85.9 ± 0.5	92.0 ± 0.5	86.2 ± 0.4	85.5 ± 0.4	91.8 ± 0.5

Note: The results illustrate the impact of each successive step of resolution, namely, cleaning, left normalization and rescuing.

Table 4. Benchmarking results for SNPs

Dataset	mpileup				GATK			
	Hours (h)	Cost ($)	Pre (%)	Rec (%)	Hours (h)	Cost ($)	Pre (%)	Rec (%)
Venter	2	5	98.5 ± 0.0	97.0 ± 0.0	46	115	98.7 ± 0.0	97.2 ± 0.0
Contaminated venter	3	8	91.1 ± 0.0	96.7 ± 0.0	57	143	98.5 ± 0.0	92.9 ± 0.0
NA12878	5	13	–	98.8 ± 0.0	97	243	–	98.8 ± 0.0
Contaminated NA12878	5	13	–	98.8 ± 0.0	90	225	–	98.8 ± 0.0
NA18507	4	10	–	99.0 ± 0.0	78	195	–	99.0 ± 0.0
NA19240	4	10	–	99.0 ± 0.0	72	180	–	99.0 ± 0.0
Mouse	6	15	98.4 ± 0.2	87.3 ± 0.2	107	268	97.8 ± 0.2	94.9 ± 0.2
Contaminated mouse	5	13	98.3 ± 0.2	86.7 ± 0.2	96	240	97.9 ± 0.2	94.6 ± 0.2

Table 5. Benchmarking results for small deletions (excluding Pindel results)

Dataset	mpileup				GATK			
	Hours (h)	Cost ($)	Pre (%)	Rec (%)	Hours (h)	Cost ($)	Pre (%)	Rec (%)
Venter	2	5	91.3% ± 0.0	75.2% ± 0.0	46	115	92.4% ± 0.0	91.3% ± 0.0
Contaminated Venter	3	8	91.7% ± 0.0	71.7% ± 0.0	57	143	92.4% ± 0.0	90.5% ± 0.0
Mouse	6	15	79.0% ± 0.4	85.9% ± 0.4	107	268	81.5% ± 0.4	95.8% ± 0.4
Contaminated mouse	5	13	80.4% ± 0.4	84.9% ± 0.4	96	240	82.8% ± 0.4	95.6% ± 0.4

identifies more than half of the deletions. Moreover, we observe that BreakDancer's recall is much higher for the sampled human datasets than for Venter and mouse. This discrepancy can be explained by the fact that, unlike the sampled NA12878 validation data, the comprehensive Venter and mouse datasets contain many short structural deletions, and BreakDancer is not designed for shorter variants.

Increasing breakpoint tolerance to 1000 bp, as reported in Supplementary Tables S8 and S11, improves both callers' precision on the synthetic and mouse datasets, more dramatically in the case of BreakDancer. Recall is only slightly improved by increasing tolerance from 100 to 1000 bp, with the notable exception being the NA19240 dataset.

We observe notably different results when evaluating the exact breakpoint of accuracy of these methods (Table 7 and Supplementary Table S9). BreakDancer makes almost no exactly accurate calls for insertions or deletions on any dataset. Pindel fares better, but loses ~2–5% of its precision and a smaller degree of recall when evaluating only variants at exact breakpoints.

Finally, we note that neither caller handles the contaminated datasets particularly well, although with somewhat different

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

Table 6. Benchmarking results for long deletions (approximate evaluation with breakpoint tolerance of 100 bp)

Dataset	Pindel				BreakDancer			
	Hours (h)	Cost ($)	Pre (%)	Rec (%)	Hours (h)	Cost ($)	Pre (%)	Rec (%)
Venter	80	200	71.8 ± 0.0	43.0 ± 0.0	2	5	7.1 ± 0.0	4.9 ± 0.0
Contaminated venter	>400	>1000	–	–	1	3	0.0 ± 0.0	0.0 ± 0.0
NA12878	81	203	–		3	8	–	95.7 ± 2.9
Contaminated NA12878	>400	>1000	–	–	>400	>1000	–	–
NA18507	168	420	–	60.0 ± 3.0	3	8	–	60.0 ± 3.0
NA19240	221	553	–	69.0% ± 2.8	29	73	–	77.6 ± 2.8
Mouse	>400	>1000	–	–	4	10	17.4 ± 1.0	15.1 ± 0.8
Contaminated mouse	345	863	74.7 ± 1.2	53.2 ± 0.8	3	8	16.7 ± 1.3	10.9 ± 0.8

Table 7. Benchmarking results for long deletions (exact evaluation)

Dataset	Pindel				BreakDancer			
	Hours (h)	Cost ($)	Pre (%)	Rec (%)	Hours (h)	Cost ($)	Pre (%)	Rec (%)
Venter	80	200	67.2 ± 0.0	40.2 ± 0.0	2	5	0.2 ± 0.0	0.2 ± 0.0
Contaminated venter	>400	>1000	–	–	1	3	0.0 ± 0.0	0.0 ± 0.0
NA12878	81	203	–	78.3 ± 2.9	3	8	–	0.0 ± 2.9
Contaminated NA12878	>400	>1000	–	–	>400	>1000	–	–
NA18507	168	420	–	60.0 ± 3.0	3	8	–	0.0 ± 3.0
NA19240	221	553	–	69.0 ± 2.8	29	73	–	5.2 ± 2.8
Mouse	>400	>1000	–	–	4	10	0.2 ± 1.0	0.1 ± 0.8
Contaminated mouse	345	863	72.5% ± 1.2	51.6 ± 0.8	3	8	0.2 ± 1.3	0.2 ± 0.8

modes of failure. Pindel suffered computationally when dealing with the contaminated datasets, failing to complete on the contaminated Venter and the contaminated NA12878 datasets, and required 375 h to process the contaminated mouse dataset. Although BreakDancer did not complete on the contaminated NA12878 dataset, it in fact executed very quickly on both the contaminated Venter and contaminated mouse datasets. However, its accuracy suffered greatly relative to its accuracy on the analogous non-contaminated datasets, as it found almost no variants in either case.

3.5 Computational performance

Although the cost of running a given caller on Amazon's AWS platform provides a convenient single metric for comparison, we also provide more fine-grained computational performance metrics to highlight differences between the callers and to help researchers optimize their choice of computational platforms. In Table 8, we present the performance metrics for all four callers on the Venter dataset (Supplementary Table S12 for statistics on other datasets). All four callers use virtually all 60.5 GB of memory available to them on Amazon's cc2.8xlarge instance; mpileup, Pindel and BreakDancer do so consistently through their runs, but GATK's memory usage fluctuates, as shown by its lower average memory usage, most likely because only portions of the GATK pipeline are multi-threaded. GATK also requires a large amount of disk space, while mpileup and Pindel use only modest amounts; as BreakDancer's output is in a more compact format than VCF, it requires almost none. We also note that because Pindel became memory-bound on our chosen instance type, we ran it single-threaded; we thus ran BreakDancer on a single thread for consistency.

4 DISCUSSION

Hundreds of variant calling algorithms have been proposed, and the majority of these algorithms have been benchmarked in some form (see detailed discussion in Section A in the Supplementary Material). To the best of our knowledge, none of these existing benchmarking methodologies accounts for noise in validation data, ambiguity in variant representation or computational efficiency of variant calling methods in a consistent and principled fashion. Given the rapid growth of next-generation sequencing data, the need for a robust and standardized methodology has never been greater.

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

Table 8. Computational performance statistics for the Venter dataset

Caller	Clock time (h)	Cost ($)	CPU time (h)	Max threads	Max disk (GB)	Max memory (GB)	Avg. memory (GB)
Mpileup	2	5	28	24	0.9	59.1	58.8
GATK	46	115	94	16	530	59.1	53.4
Pindel	80	200	80	1	3.4	59.1	58.7
BreakDancer	2	5	2	1	0.02	59.1	58.7

The IT industry serves as an illuminating case study in the context of benchmarking. Similar to next-generation sequencing technology, the IT industry has benefited greatly from the additional hardware resources provided by Moore's Law. However, the industry's rapid progress also hinged on the agreement on proper metrics to measure performance as well as consensus regarding the best benchmarks to run to fairly evaluate competing systems (Patterson, 2012). Prior to this industry-wide agreement, each company invented its own metrics and ran its own set of benchmarking evaluations, making the results incomparable and customers suspicious of them. Even worse, engineers at competing companies were unable to determine the usefulness of their competitors' innovations, and so the competition to improve performance occurred only within companies rather than between them. Once the IT industry agreed on a fair playing field, progress accelerated, as engineers could see which ideas worked well (and which did not), and new techniques were developed to build on promising approaches.

Similarly, we believe that SMaSH could help accelerate progress in the field of genomic variant calling. We have compiled a rich collection of datasets and developed a principled set of evaluation metrics that together allows for quantitative evaluation of variant calling algorithms in terms of accuracy, computational efficiency and robustness/ease-of-use (via ability to run on AWS). Moreover, although SMaSH currently focuses on benchmarking variant calling algorithms for normal human genomes, we believe that the motivating ideas behind SMaSH, along with the tools developed as part of SMaSH, will be useful in devising analogous variant calling benchmarking toolkits for human cancer genomes and for the genomes of other organisms.

Finally, we view SMaSH as a work in progress, as the contents of SMaSH reflect (and are limited by) existing technologies. SMaSH currently has limited ground truth data for human genomes, and the validation data across datasets are enriched in 'easier' non-repetitive regions owing to underlying sequencing and chip biases. Like any benchmarking suite, SMaSH must evolve over time to stay relevant. As new sources of validation data become available, e.g. the NA12878 knowledge base (M.DePristo, 2013, personal communication) or curated variants from Illumina's Platinum Genome, these datasets should be incorporated into SMaSH. Existing datasets should also be updated to keep them fresh and prevent algorithms from 'overfitting' to stale benchmarks, and benchmarking datasets should be deprecated as new data sources obviate their utility.

ACKNOWLEDGEMENTS

The authors thank David Bentley, Bill Bolosky, Mauricio Carneiro, Mark Depristo, Michael Eberle, Adam Ewing, Gaddy Getz, David Haussler, Jeff Kidd, Jon Kuroda, Elliott Margulies, Jim Mullikin, Frank Nothaft, Ravi Pandya, Benedict Paten, Taylor Sittler, Arun Wiita, Kai Ye and Matei Zaharia for useful insights.

Funding: This research was supported in part by NSF awards (1122732 and DBI-0846015), NIH National Research Service Award Trainee appointment (T32-HG00047), NSF CISE Expeditions award (CCF-1139158), DARPA XData Award (FA8750-12-2-0331) and gifts from Amazon Web Services, Google, SAP, Cisco, Clearstory Data, Cloudera, Ericsson, Facebook, FitWave, General Electric, Hortonworks, Huawei, Intel, Microsoft, NetApp, Oracle, Samsung, Splunk, VMware, WANdisco and Yahoo!.

Conflict of interest: none declared.

REFERENCES

Albers,C.A. *et al.* (2011) Dindel: accurate indel calls from short-read data. *Genome Res.*, **21**, 961–973.

Alkan,C. *et al.* (2011) Genome structural variation discovery and genotyping. *Nat. Rev. Genet.*, **12**, 363–376.

Chen,K. *et al.* (2009) BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nat. Methods*, **6**, 677–681.

Church,D.M. *et al.* (2009) Lineage-specific biology revealed by a finished genome assembly of the mouse. *PLoS Biol.*, **7**, e1000112.

The 1000 Genomes Project Consortium. (2010) A map of human genome variation from population-scale sequencing. *Nature*, **467**, 1061–1073.

DePristo,M. *et al.* (2011) A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.*, **43**, 491–498.

Earl,D. *et al.* (2011) Assemblathon 1: a competitive assessment of de novo short read assembly methods. *Genome Res.*, **21**, 2224–2241.

Frazer,K.A. *et al.* (2007) A second generation human HAPlotype map of over 3.1 million SNPs. *Nature*, **449**, 851–861.

Gnerre,S. *et al.* (2011) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl Acad. Sci. USA*, **108**, 1513–1518.

The HapMap Consortium. (2010) Integrating common and rare genetic variation in diverse human populations. *Nature*, **467**, 52–58.

Kedes,L. and Campany,G. (2011) The new date, new format, new goals and new sponsor of the archon genomics x PRIZE competition. *Nat. Genet.*, **43**, 1055–1058.

Kidd,J.M. *et al.* (2010a) Characterization of missing human genome sequences and copy-number polymorphic insertions. *Nat. Methods*, **7**, 365–371.

Kidd,J.M. *et al.* (2010b) A human genome structural variation sequencing resource reveals insights into mutational mechanisms. *Cell*, **143**, 837–847.

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016

Levy,S. *et al.* (2007) The diploid genome sequence of an individual human. *PLoS Biol.*, **5**, e254.
Li,H. *et al.* (2009) The sequence alignment/map (sam) format and samtools. *Bioinformatics*, **25**, 2078–2079.
Lyon,G. *et al.* (2012) Low concordance of variant calling algorithms in exome sequencing. In: *Meeting of The American Society of Human Genetics*. San Francisco, CA.
Mardis,E.R. (2010) The $1,000 genome, the $100,000 analysis? *Genome Med.*, **2**, 84.
Nekrutenko,A. and Taylor,J. (2012) Next-generation sequencing data interpretation: enhancing reproducibility and accessibility. *Nat. Rev. Genet.*, **13**, 667–672.
Patterson,D. (2012) For better or worse, benchmarks shape a field: technical perspective. *Commun. ACM.*, **55**, 104.
Yalcin,B. *et al.* (2011) Sequence-based characterization of structural variation in the mouse genome. *Nature*, **477**, 326–329.
Ye,K. *et al.* (2009) Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads. *Bioinformatics*, **25**, 2865–2871.
Zook,J.M. and Salit,M. (2011) Genomes in a bottle: creating standard reference materials for genomic variation - why, what and how? *Genome Biol.*, **12**, P31.

Downloaded from http://bioinformatics.oxfordjournals.org/ at :: on August 30, 2016