Genotyping structural variation in variation graphs with the vg toolkit

This manuscript (permalink) was automatically generated from jmonlong/manu-vgsv@453702b on May 8, 2019.

Authors

Glenn Hickey[1,☯], David Heller[1,2,☯], Jean Monlong[1,☯], Jonas Andreas Sibbesen[1], Jouni Siren[1], Jordan Eizenga[1], Eric Dawson[3], Erik Garrison[1], Adam Novak[1], Benedict Paten[1,†]

[☯] — These authors contributed equally to this work

[†] — To whom correspondence should be addressed: bpaten@ucsc.edu

1. UC Santa Cruz Genomics Institute, University of California, Santa Cruz, California, USA
2. Max Planck Institute for Molecular Genetics, Berlin, Germany
3. Department of Genetics, University of Cambridge, Cambridge, UK

Abstract

Structural variants (SVs) are significant components of genetic diversity and have been associated with diseases, but the technological challenges surrounding their representation and identification make them difficult to study relative to point mutations. Still, thousands of SVs have been characterized, and catalogs continue to improve with new technologies. In parallel, variation graphs have been proposed to represent human pan-genomes, offering reduced reference bias and better mapping accuracy than linear reference genomes. We contend that variation graphs provide an effective means for leveraging SV catalogs for short-read SV genotyping experiments. In this work, we extend vg (a software toolkit for working with variation graphs) to support SV genotyping. We show that it is capable of genotyping insertions, deletions and inversions, even in the presence of small errors in the variant representation. We then benchmark vg against state-of-the-art SV genotypers using three high-quality sequence-resolved SV catalogs generated by recent studies. Our results show that vg produced the best genotype predictions systematically in all datasets. In addition, we use assemblies from 12 yeast strains to show that graphs constructed directly from aligned de novo assemblies can improve genotyping compared to graphs built from intermediate SV catalogs in the VCF format. Our results demonstrate the power of variation graphs for SV genotyping. Beyond single nucleotide variants and short insertions/deletions, the vg toolkit now incorporates SVs in its unified variant calling framework and provides a natural solution to integrate high-quality SV catalogs and assemblies.

Introduction

A structural variant (SV) is a genomic mutation involving 50 or more base pairs. SVs can take several forms such as deletions, insertions, inversions, translocations or other complex events. Due to their greater size, SVs often have a larger impact on phenotype than smaller events such as single nucleotide variants (SNVs) and small insertions and deletions (indels). Indeed, SVs have long been associated with developmental disorders, cancer and other complex diseases and phenotypes[1].

Despite their importance, SVs remain much more poorly studied than their smaller mutational counterparts. This discrepancy stems from technological limitations: Short read sequencing has provided the basis of most modern genome sequencing studies due to its high base-level accuracy and relatively low cost, however, it is poorly suited for discovering SVs. The central obstacle is in mapping short reads to the human reference genome. It is generally difficult or impossible to unambiguously map a short read if the sample whose genome is being analyzed differs substantially from the reference at the read's location. The large size of SVs virtually guarantees that this will be the case. For example, if the read derives from the middle of a large insertion relative to the reference, there is no sequence in the reference that corresponds to a correct mapping. The best result a read mapper could hope to produce would be to leave it unmapped. Moreover, SVs often lie in repeat-rich regions, which further frustrate read mapping algorithms.

Short reads can be more effectively used to genotype known SVs. This is important, as even though efforts to catalog SVs with other technologies have been highly successful, their cost currently prohibits their use in large-scale studies that require hundreds or thousands of samples such as disease association studies. Traditional SV genotypers start from reads that were mapped to a reference genome, extracting aberrant mapping that might support the presence of the SV of interest. State-of-art methods like SVTyper[2] and Delly[3] typically focus on split reads and paired reads mapped too close or too far from each other. These discordant reads are tallied and remapped to the reference sequence modified with the SV of interest in order to genotype deletions, insertions, duplications, inversions and translocations. SMRT-SV2 uses a different approach: the reference genome is augmented with SV-containing sequences as alternate contigs and the resulting mappings are evaluated with a machine learning model trained for this purpose[4].

The catalog of known SVs in human is quickly expanding. Several large-scale projects have used short-read sequencing and extensive discovery pipelines on large cohorts, compiling catalogs with tens of thousands of SVs in humans[5,6]. More recent studies using long-read or linked-read sequencing have produced large catalogs of structural variation, the majority of which was novel and sequence-resolved[10,4,7,8,9]. These technologies are also enabling the production of high-quality de novo genome assemblies[11,7], and large blocks of haplotype-resolved sequences[12]. Such technical advances promise to expand the amount of known genomic variation in humans in

the near future, and further power SV genotyping studies. Representing known structural variation in the wake of increasingly larger datasets poses a considerable challenge, however. Standard formats such as VCF are unwieldy when used for SVs due to ambiguity in their specification and limited support across tools. Another strategy consists in incorporating SVs into a linear pan-genome reference via alt contigs, but it also has serious drawbacks. Alt contigs tend to increase mapping ambiguity. In addition, it is unclear how to scale this approach as SV catalogs grow.

Pan-genome graph reference representations offer an attractive approach for storing genetic variation of all types[13]. The graph structure can represent SVs as succinctly and directly as point mutations. Moreover, including known variants in the reference makes both read mapping and variant calling variant-aware. This leads to benefits in terms of accuracy and sensitivity[14,15,16]. In addition, different variant types can be called simultaneously and scored consistently across types by a unified framework.

vg is the first openly available variation graph tool to scale to multi-gigabase genomes. It provides read mapping, variant calling and visualization tools[14]. In addition, vg can build graphs both from variant catalogs in the VCF format and from assembly alignments.

Other tools have used genome graphs specifically to genotype variants. GraphTyper realigns mapped reads to a graph built from known SNVs and short indels using a sliding-window approach[17]. BayesTyper first builds a set of graphs from known variants including SVs, then genotypes variants by comparing the distribution k-mers in the sequencing reads with the k-mers of putative haplotype paths in the graph[18]. These graph-based approaches showed clear advantages over standard methods that use only the linear reference.

In this work, we present a variation graph-based SV genotyping framework implemented using vg. We show that this method is capable of genotyping known deletions, insertions and inversions. On simulated variants, vg is robust to small errors in the breakpoint location, and it outperforms most other methods on shallow sequencing experiments. Starting from SVs discovered in recent long-read sequencing studies[19,20,21,4], we evaluated the genotyping accuracy using simulated and real Illumina reads. We also compared vg's performance with state-of-the-art SV genotypers: SVTyper[2], Delly[3], BayesTyper[18] and SMRT-SV2[4]. Across all three datasets that we tested, vg is the best performing SV genotyper on real short-read data for all SV types. In addition, we show that building graphs from the alignment of de novo assemblies leads to better genotyping performance.

Results

Structural variation in vg

In vg, a variation graph represents DNA sequences and genomic variation using sequence nodes linked by edges (see Supplementary Information and S1). We used vg to implement a simple SV

genotyping pipeline. Reads are mapped to the graph and used to compute the read support for each node and edge. Sites of variation are then identified using the snarl decomposition as described in [22]. For each site, the two most supported paths (haplotypes) are determined, and their relative supports used to produce a genotype at that site (Figure 1a). The pipeline is described in more detail in Methods. We rigorously evaluated the accuracy of our method on a variety of datasets, and the results are presented in the remainder of this section.

Simulated dataset

As a proof-of-concept we simulated genomes and different types of SVs with a size distribution matching real SVs[19]. We compared vg against SVTyper, Delly and BayesTyper across different levels of sequencing depth. Some errors were also added at the breakpoints to investigate their effect on genotyping accuracy (see Methods). The results are shown in Figure 1b. When using the correct breakpoints, vg tied with Delly as the best genotyper for deletions, and with BayesTyper as the best genotyper for insertions. For inversions, vg was the second best genotyper after BayesTyper. The differences between the methods were the most visible at lower sequencing depth. In the presence of 1-10 bp errors in the breakpoint locations, the performance of Delly and BayesTyper dropped significantly (Figure 1b). The dramatic drop for BayesTyper can be explained by its k-mer-based approach that requires precise breakpoints. In contrast, vg was only slightly affected by the presence of errors. For vg, the F1 scores for all SV types decreased no more than 0.07. Overall, these results show that vg is capable of genotyping SVs and is robust to breakpoint inaccuracies in the input VCF.

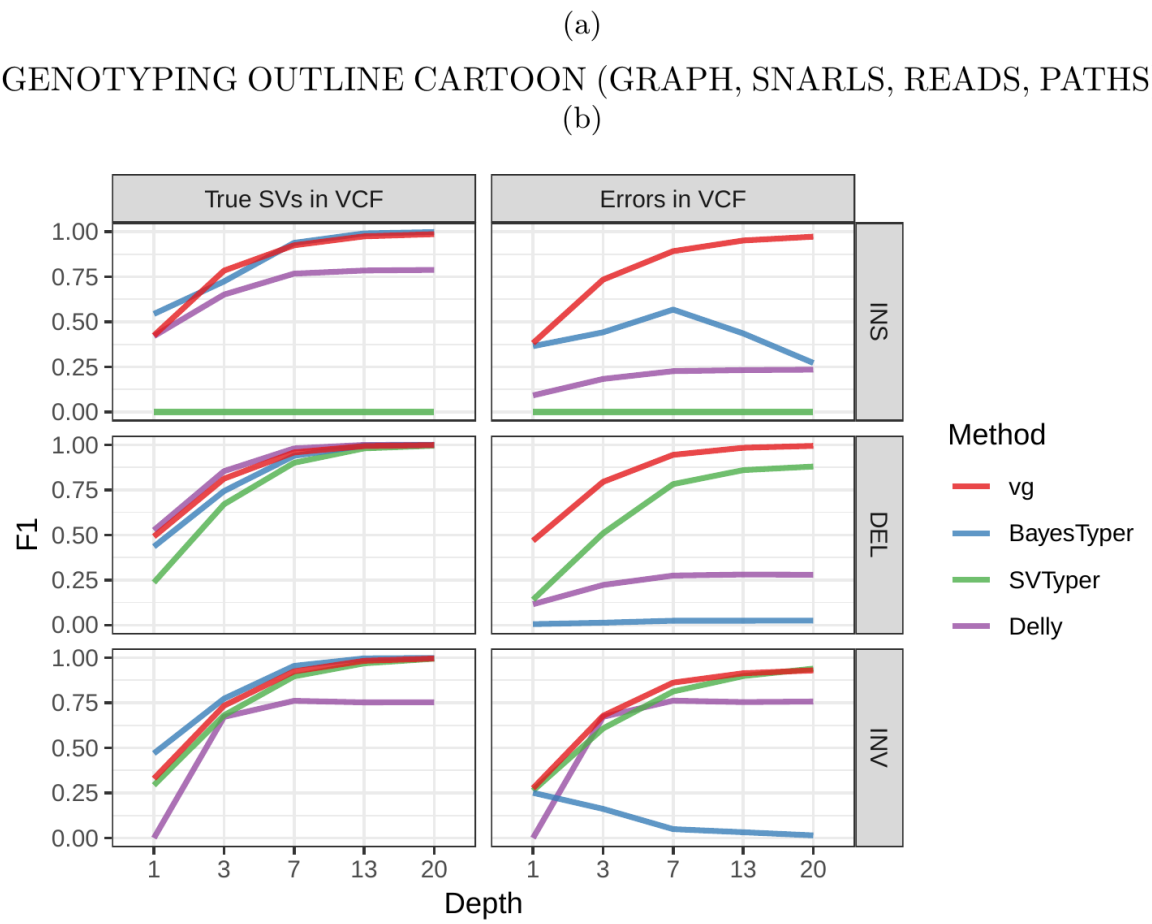


Figure 1: **Structural variation in vg.** a) Adding large insertions, deletions and inversions in a variation graph. b) Simulation experiment. For each experiment (method, depth and input VCF with/without breakpoint errors), the y-axis shows the maximum F1 across different minimum quality thresholds.

HGSVC dataset

Structural variants from The Human Genome Structural Variation Consortium (HGSVC) were used to benchmark the genotyping performance of vg against 3 other SV genotyping methods This high-quality SV catalog was generated from three samples using a consensus from different sequencing, phasing and variant calling technologies[19]. The three samples come from different human populations: a Han Chinese individual (HG00514), a Puerto-Rican individual (HG00733), and a Yoruban Nigerian individual (NA19240). These SVs were used to construct a graph with vg and as input for the other genotypers. Using short sequencing reads, the SVs were genotyped and compared with the genotypes in the original catalog (see Methods).

First, by simulating reads for HG00514, we compared the different methods in the ideal error-free situation where the SV catalog matches exactly the SVs supported by the reads. While vg outperformed Delly and SVTyper, BayesTyper showed the best F1 score and precision-recall trade-off (Figures 2 and S2, Table S1). When restricting the comparisons to regions not identified as

tandem repeats or segmental duplications, the genotyping predictions were significantly better for all methods, with vg almost as good as BayesTyper on deletions (F1 of 0.944 vs 0.955). We observed similar results when evaluating the presence of an SV call instead of the exact genotype (Figures 2 and S3). Overall, both graph-based methods, vg and BayesTyper, outperformed the other two methods tested.

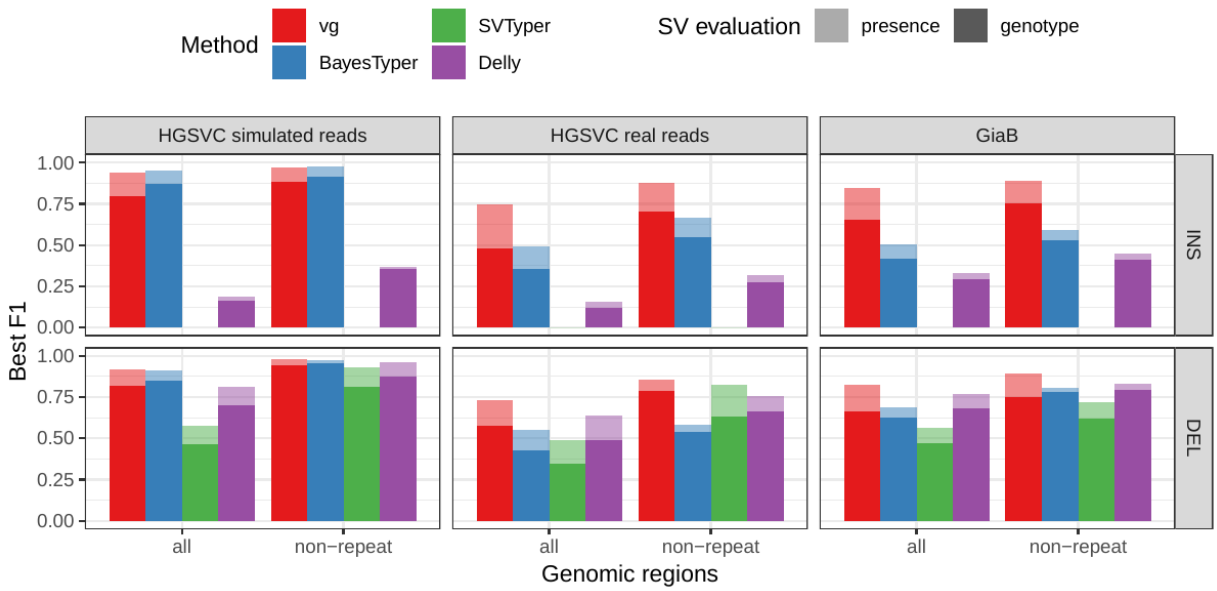
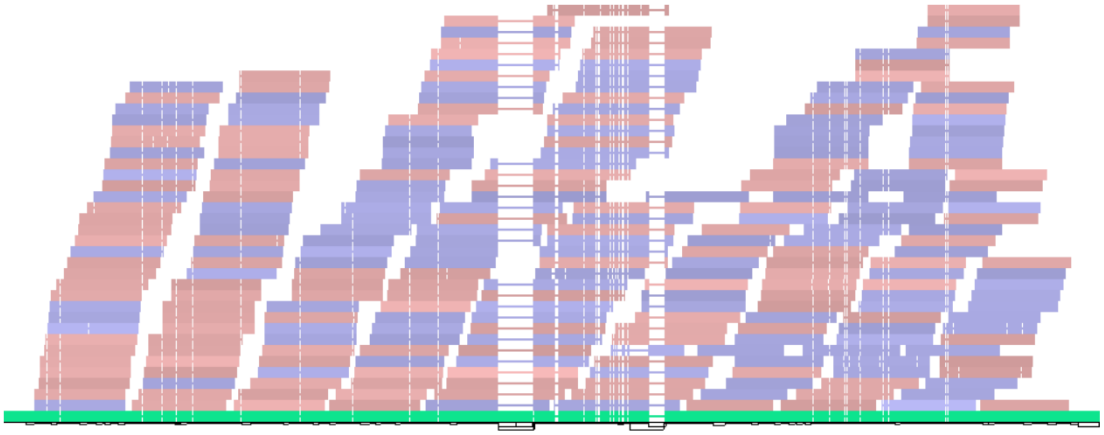


Figure 2: **Structural variants from the HGSVC and Genome in a Bottle datasets**. HGSVC: Simulated and real reads were used to genotype SVs and compared with the high-quality calls from Chaisson et al.[19]. Reads were simulated from the HG00514 individual. Using real reads, the three HG00514, HG00733, and NA19240 individuals were tested. GiaB: Real reads from the HG002 individual were used to genotype SVs and compared with the high-quality calls from the Genome in a Bottle consortium[20,21]. Maximum F1 score for each method (color), across the whole genome or focusing on non-repeat regions (x-axis). We evaluated the ability to predict the presence of an SV (transparent bars) and the exact genotype (solid bars). SVTyper cannot genotype insertions hence the absent bars in the top panel.

We then repeated the analysis using real Illumina reads from HG00514 to benchmark the methods on a more realistic experiment. Here vg clearly outperformed other approaches, most likely because of its graph-based strategy and robustness to errors in the SV catalog (Figures 2 and S4). In non-repeat regions and across the whole genome, the F1 scores and precision-recall AUC were higher for vg compared to other methods. For example, for deletions in non-repeat regions, the F1 score for vg was 0.801 while the second best method, Delly, had a F1 score of 0.692. We observed similar results when evaluating the presence of an SV call instead of the exact genotype (Figures 2 and S5). Figure 3 shows examples of an exonic deletion and an exonic insertion that were correctly genotyped by vg but not by the other methods.

(a)

chr2:100282798-100282848



(b)

chr12:116277336-116277336

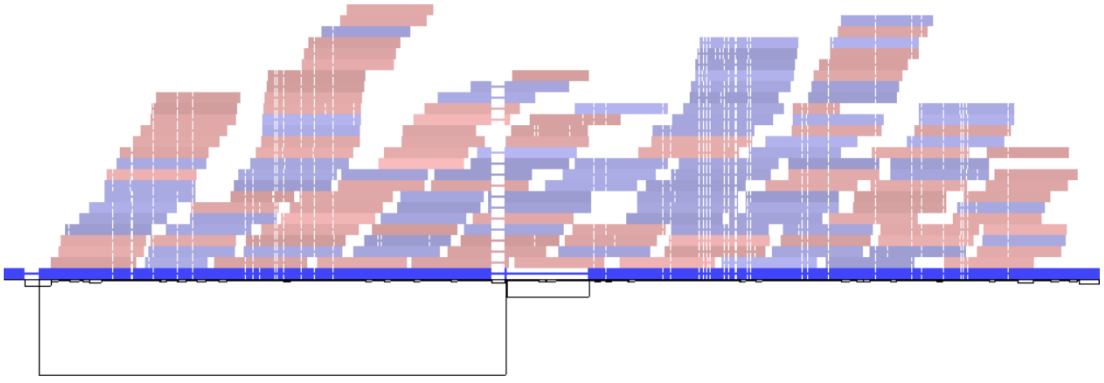


Figure 3: **Exonic SVs in the HGSVC dataset correctly genotyped by vg**. The red/blue segments represent read mapped to the graph whose topology is shown at the bottom in black. Between the graph and the reads, a colored horizontal bar shows the path followed by the reference genome (GRCh38). a) 51 bp homozygous deletion in the last exon of the LONRF2 gene. b) 114 bp homozygous insertion in a short tandem repeat region overlapping the first exon of the MED13L gene, a gene predicted to be loss of function intolerant.

Other long-read datasets

The Genome in a Bottle (GiaB) consortium is currently producing a high-quality SV catalog for a Ashkenazim individual (HG002)[20,21]. Dozens of SV callers and datasets from short, long and linked reads were used to produce this set of SVs. vg performed similarly on this dataset as on the HGSVC dataset, with a F1 score of 0.75 for both insertions and deletions in non-repeat regions (Figures 2, S6 and S7, and Table S2). As before, other methods produced lower F1 scores in most cases, although Delly and BayesTyper predicted better genotypes for deletions in non-repeat regions.

A recent study by Audano et al. generated an SV catalog using long-read sequencing across 15 individuals[4]. These variants were then genotyped from short reads across 440 individuals using SMRT-SV2, a machine-learning genotyper implemented for this study. SMRT-SV2 was trained on a pseudo-diploid genome constructed from high quality assemblies of two haploid cell lines. We first called SVs in this dataset, using the same SV catalog and short read dataset. vg was systematically better at predicting the presence of an SV for both SV types, but SMRT-SV2 produced better genotypes for deletions (see Figures 4, S8 and S9, and Table S3). Using publicly available Illumina reads, we then genotyped SVs in 3 of the 15 individuals that were used for discovery in Audano et al.[4]. Compared to SMRT-SV2, vg had a better precision-recall curve and a higher F1 for both insertions and deletions (SVPOP in Figures 4 and S10, and Table S4). Of note, SMRT-SV2 produces *no-calls* in regions where the read coverage is too low, and we observed that its recall increased when filtering these regions out the input set. Interestingly, vg performed well even in regions where SMRT-SV2 produced *no-calls* (Figure S11 and Table S5). Finally, Audano et al. identified 217 sequence-resolved inversions. vg correctly predicted the presence of around 14% of the inversions present in the three samples (Table S4). Inversions are often complex, harboring additional variation that makes their characterization and genotyping challenging.

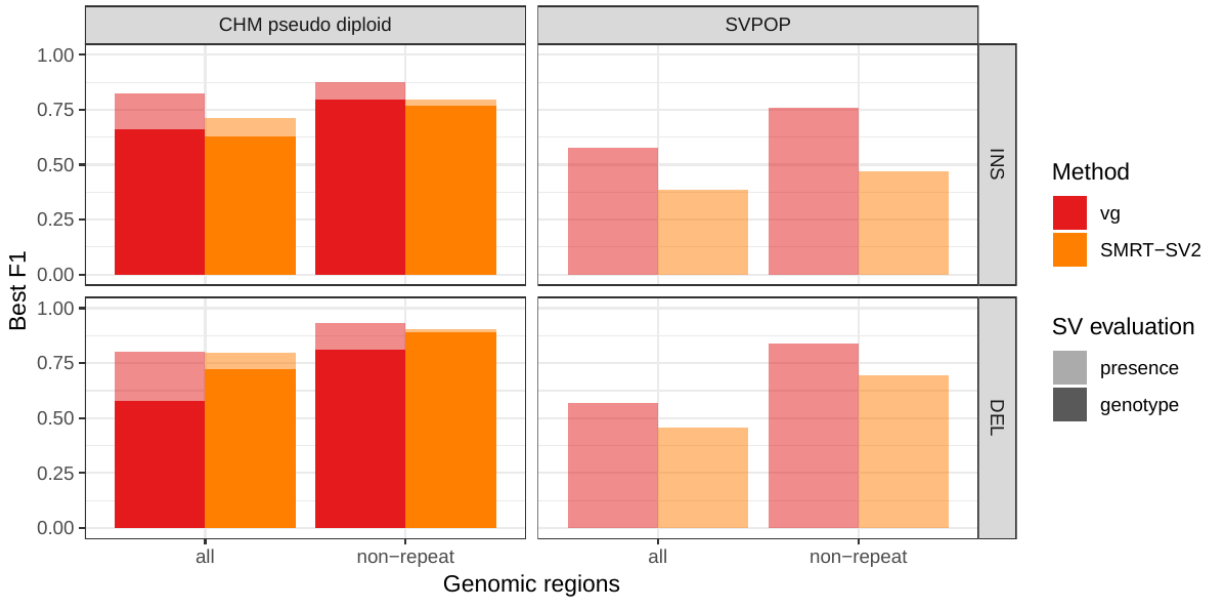


Figure 4: **Structural variants from Audano et al.[4]**. The pseudo-diploid genome built from CHM cell lines was originally used to train SMRT-SV2 in Audano et al.[4]. The SVPOP panel shows the combined results for the HG5014, HG00733 and NA19240 individuals, 3 of the 15 individuals used to generate the high-quality SV catalog in Audano et al.[4]. Maximum F1 score for each method (color), across the whole genome or focusing on non-repeat regions (x-axis). We evaluated the ability to predict the presence of an SV (transparent bars) and the exact genotype (solid bars). Genotype information is not available in the SVPOP catalog hence genotyping performance could not be evaluated.

Graphs from alignment of de novo assemblies

Genome graphs can be constructed directly from multiple sequence alignments of de novo assemblies[14]. This bypasses the need for generating an explicit variant catalog relative to a linear reference which could be a source of error (for example, from reference bias during read mapping and variant calling). Furthermore, genome alignments from software such as Cactus [23] can contain complex structural variation that is extremely difficult to represent, let alone call, outside of a graph. We therefore investigated whether genome graphs derived from de-novo assembly alignments yield advantages for SV genotyping. To this end, we analyzed public sequencing datasets for 12 yeast strains from two related clades (*S. cerevisiae* and *S. paradoxus*) [24]. We generated and compared two different types of genome graphs using 5 of the 12 strains, *S.c. S288C* as the reference strain and two other strains for each yeast clade (see Methods). The first graph type (in the following called *VCF graph*) was created from the linear reference genome of the *S.c. S288C* strain and a set of SVs relative to this reference strain in VCF format identified by three methods: Assemblytics [25], AsmVar [26] and paftools [27]. The second graph (in the following called *cactus graph*) was derived from a multiple genome alignment of the five strains using our Cactus tool [23]. The *VCF graph* is mainly linear and highly dependent on the reference genome. In contrast, the *cactus graph* is structurally complex and free of reference bias.

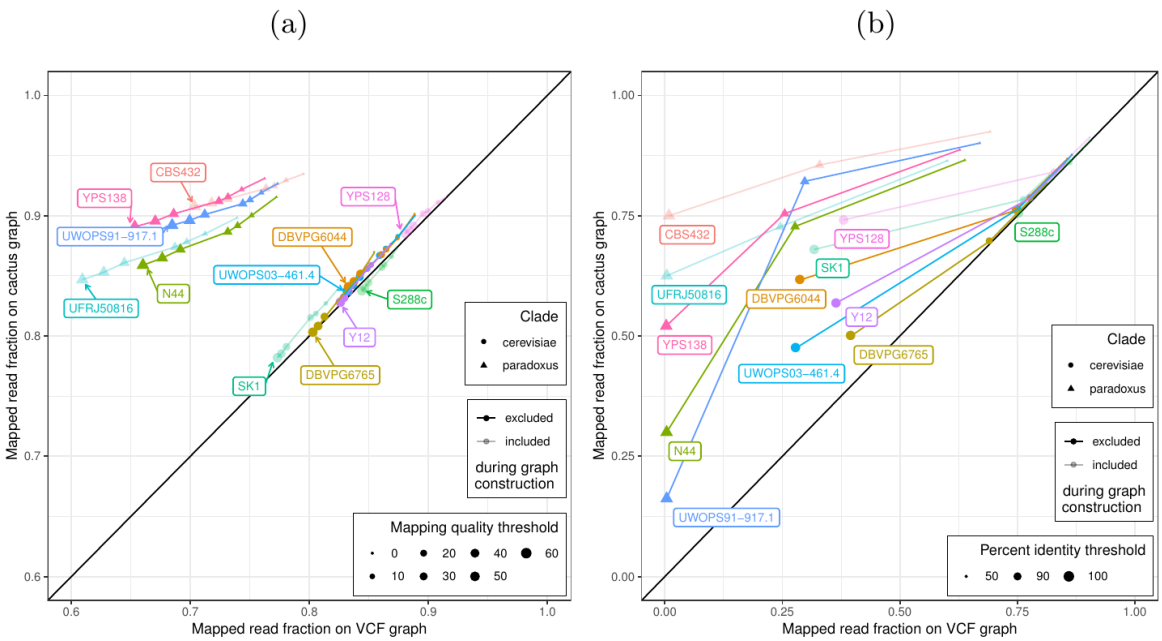


Figure 5: **Mapping comparison.** Short reads from all 12 yeast strains were aligned to both graphs. The fraction of reads mapped to the cactus graph (y-axis) and the VCF graph (x-axis) are compared. a) Stratified by mapping quality threshold. b) Stratified by percent identity threshold. Colors and shapes represent the 12 strains and two clades, respectively. Transparency indicates whether the strain was included or excluded in the graphs.

First, we tested our hypothesis that the *cactus graph* has higher mappability due to its better representation of sequence diversity among the yeast strains. Figure 5a shows the fraction of Illumina reads from the 12 strains that was mapped with a mapping quality above a certain threshold to the *cactus graph* and to the *VCF graph*. Generally, more reads were mapped to the *cactus graph* than to the *VCF graph* regardless of the chosen mapping quality threshold. Only for the reference strain *S.c. S288C*, the *VCF graph* exhibited slightly better mappability. This suggests that the improvement in mappability is not driven by the higher sequence content in the *cactus graph* alone (15.4 Mb compared to 12.4 Mb in the *VCF graph*). Instead, our measurements suggest that the genetic distance to the reference strain correlates with the better mapping on the *cactus graph* over the *VCF graph* (Pearson's product-moment correlation, p-value=1.415e-11). Consequently, the benefit of using the *cactus graph* is largest for strains in the *S. paradoxus* clade and smaller for reads from strains in the *S. cerevisiae* clade. We observed a similar trend when exploring the mapping identity of the short reads on the graphs (see Figure 5b and Supplementary Information).

Interestingly, our measurements did not show a substantial difference between strains included in the graph and excluded strains. The results suggest that two strains from each clade as well as the reference strain are sufficient to capture most of the genetic variation among all the strains. Only the number of alignments with perfect identity is substantially lower for the strains that were not

included in the creation of the graphs (see Figure 5b). For a direct comparison, see Figure S13 which shows results of the same experiment on graphs generated from all 12 strains.

Next, we compared the SV genotype performance of both graphs. We mapped short reads from the 11 non-reference strains to both graphs and called variants for each strain using the vg toolkit's variant calling module (see Methods). In the absence of a gold standard, we evaluated each callset based on the alignment of reads to a *sample graph* constructed from the callset (see Methods). If a given callset is correct, we expect that reads from the same sample will be mapped confidently and with high identity to the corresponding sample graph. Therefore, we compared the average mapping quality and percent identity of the short reads on each sample graph (see Figures 6a and b). Similar to the results of our mapping analysis above, the *cactus graph* clearly outperformed the *VCF graph* for strains in the *S. paradoxus* clade and performed slightly better for strains in the *S. cerevisiae* clade. Again, our measurements did not show a large difference between strains included in the graph and those that were excluded. For a direct comparison, see Figure S14 which shows results of the same experiment on graphs generated from all 12 strains.

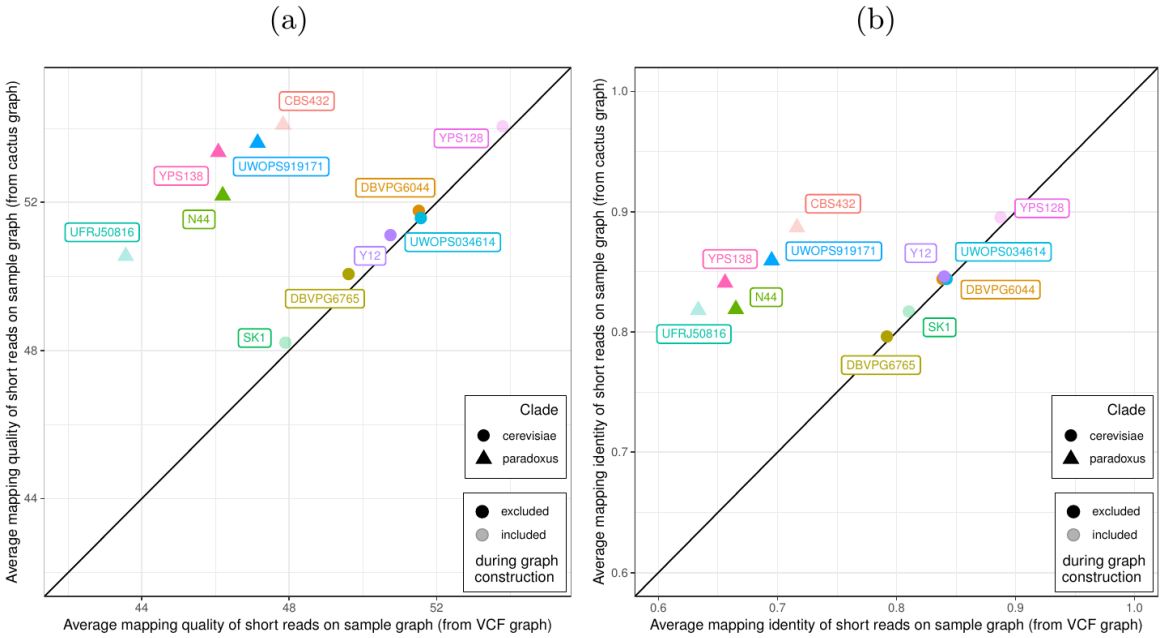


Figure 6: **SV genotyping comparison.** Short reads from all 11 non-reference yeast strains were used to genotype SVs contained in both graphs. Subsequently, sample graphs were generated from the resulting SV callsets. The short reads were again aligned to the sample graphs and the quality of the alignments was used to ascertain genotyping performance. a) Average mapping quality of short reads aligned to the sample graphs derived from *cactus graph* (y-axis) and *VCF graph* (x-axis). b) Average mapping identity of short reads aligned to the sample graphs derived from *cactus graph* (y-axis) and *VCF graph* (x-axis). Colors and shapes represent the 11 non-reference strains and two clades, respectively. Transparency indicates whether the strain was included or excluded in the graphs.

Discussion

vg was the most accurate SV genotyper in our benchmarks, overall. These results show that variant calling benefits from variant-aware read mapping, a finding consistent with previous graph-based studies[14,15,16,17,18]. In addition to its variant-aware read mapping, vg offers additional advantages. Its unified framework calls and scores different variant types simultaneously. For variant interpretation in particular, a comprehensive and unified characterization of the genomic variation will be extremely valuable. Variation graphs that contain known SNVs, indels and SVs could serve as a richer reference for large scale projects that use short-read sequencing. More and more large-scale projects are sequencing the genomes of thousands or hundreds of thousands of individuals and could benefit from such a framework, e.g. the Pancancer Analysis of Whole Genomes[28], the Genomics England initiative[29], and the TOPMed consortium[30]. In contrast to incorporating SVs into a linear pan-genome reference via alt contigs, the graph structure can represent SVs as succinctly as possible. Alt contigs often contain redundant sequence resulting in increased mapping ambiguity and involves custom pipelines and non-standard metadata formats linking the contigs back to the reference. These issues make the alt contigs difficult to use, maintain, and scale as SV catalogs continue to expand.

Our method requires near-breakpoint resolution in the variant library used to construct the graph. Simulations have shown that SV genotyping with vg is robust to errors of as much as 10 bp in breakpoint location. Variants with higher uncertainty in the breakpoint location, for example discovered through read coverage analysis, cannot be safely added to the graph. By being robust to some errors in the breakpoint location, vg was more accurate in real data compared to the other genome graph method tested, BayesTyper, which assumes sequence-resolved variants as input. Of note, vg is also capable of fine-tuning SV breakpoints using an augmentation step that modifies the graph based on the read alignment. While this augmentation approach was developed to discover novel SNVs and indels, simulations showed that it is capable of correcting erroneous SV breakpoints (Figure S12 and Table S6).

The vg toolkit is under active development. Read mapping is an area of constant improvement, in terms of both computational efficiency and accuracy. One technique under development is the application of haplotype information for the improvement of read mapping and variant calling. We believe that this technique stands to benefit SV genotyping with vg, as haplotype information might enable inference at the scale of SVs when using short reads.

In our benchmarks, other methods were superior in a handful datasets and situations, primarily when genotyping deletions. However, even in most of these cases, vg had the best accuracy when evaluating only the presence or absence of each variant call. This suggests that the performance shortfall can be attributed to the genotyping algorithm rather than the mapping pipeline. We hope to address these issues in a future release.

Our results suggest that constructing a graph from de novo assembly alignment instead of a VCF leads to better SV genotyping. High quality de novo assemblies for human are becoming more and more common, for example from optimized mate-pair libraries[31] or long-read sequencing[11]. For an optimal representation of genomic variation, we expect the future graphs to include information from the alignment of numerous de novo assemblies. We are presently working on scaling our pipeline to human-sized genome assemblies. Aligning assembled contigs to existing variation graphs, like to ones created from SVs catalogs, is still experimental but could generate a genome graph augmented with both existing variant databases and new high-quality assemblies.

Conclusion

In this study, the vg toolkit was compared to existing SV genotypers across several high-quality SV catalogs. We showed that its implementation of variation graphs lead to a better SV genotyping compared to methods that rely on read mapping to a linear reference genome or a variation graph approach that requires exact sequence-resolved variants. This work introduces a flexible strategy to integrate the growing number of SVs being discovered with higher resolution technologies into the unified framework of variation graphs. This study also shows the benefit of starting directly from de novo assemblies rather than variant catalogs to integrate SVs in genome graphs.

Methods

toil-vg

toil-vg is a set of Python scripts for simplifying vg tasks such as graph construction, read mapping and SV genotyping. It uses the Toil workflow engine [32] to seamlessly run pipelines locally, on clusters or on the cloud. All variation graph analysis in this report was done using toil-vg, with the exact commands available at github.com/vgteam/sv-genotyping-paper. The principal toil-vg commands used are described below.

toil-vg construct

toil-vg construct automates graph construction and indexing following the best practices put forth by the vg community. Graph construction is parallelized across different sequences from the reference FASTA, and different whole-genome indexes are created side by side when possible. Phasing information from the input VCF can be used when available to preserve haplotypes in the GCSA2 pruning step, as well as to extract haploid sequences to simulate from.

toil-vg map

toil-vg map splits the input reads into batches, maps each batch in parallel, then merges the result.

toil-vg call

A simple though very general variant caller has been implemented as `vg call`. Here it is used to genotype structural variants already present in the graph, but the same algorithm can also be used for smaller variants such as SNPs, as well as making de-novo calls. The algorithm is as follows:

1. The average read support for each node and edge, adjusted for mapping and base quality, is computed. The graph can optionally be augmented to include new variation from the reads using a support cutoff.
2. The graph is then decomposed into snarls[22]. Briefly, a snarl is a subgraph defined by two end nodes, where cutting the graph at these nodes disconnects the snarl from the rest of the graph. Snarls can be nested inside other snarls, and this nesting hierarchy forms a forest. As proposed in Paten et al.[22], we use the snarl decomposition as a structure for identifying variants in a graph.
3. Root-level snarls from the decomposition are considered independently and in parallel. Only snarls whose two ends lie on a reference (i.e. chromosome) path are considered as the VCF format used for output requires reference positions. The following steps are performed on each root snarl.
    1. A set of paths between the snarls end nodes are computed using a heuristic search that enumerates paths until all nodes and edges in the snarl are contained in at least one path.
    2. The paths are ranked according to their average support from the reads.
    3. A genotype is determined using the relative support of the best paths, as well as the background read depth. The same logic is used for all types of variation, each of which can be expressed simply as a path in the graph.
    4. The VCF variants are derived from the paths.

Due to the high memory requirements of the current implementation of vg call, toil-vg call splits the input graph into 2.5Mb overlapping chunks along the reference path. Each chunk is called independently in parallel and the results are concatenated into the output VCF.

toil-vg sveval

The variants are first normalized with `bcftools norm` (1.9) to ensure consistent representation between called variants and baseline variants[33]. We then implemented an overlap-based strategy to compare SVs and compute evaluation metrics (sveval R package: https://github.com/jmonlong/sveval).

For deletions and inversions, the affected regions in the reference genome are overlapped and matched between the two sets of SVs. First, we select pairs of SVs with at least 10% reciprocal overlap. Then for each variant we compute the proportion of its region that is covered by an overlapping variant in the other set. If this coverage proportion is higher than 50%, the variant is

considered *covered*. True positives are covered variants from the call set or the truth set. False positives are variants from the call set that are not covered (by the truth set). False negative are variants from the truth set that are not covered (by the call set).

For insertions, we select pairs of insertions that are located no farther than 20 bp from each other. We then align the inserted sequences using a Smith-Waterman alignment. For each insertion we compute the proportion of its inserted sequence that aligns a matched variant in the other set. As for deletions/inversions, this coverage proportion is used to annotate variants as true positives, false positives and false negatives.

sveval accepts VCF files with symbolic or explicit representation of the SVs. If the explicit representation is used, multi-allelic variants are split and their sequences right-trimmed. When inversions are considered, the reverse-complement of the ALT sequence of variants larger than 10 bp is aligned to the REF sequence and classified as an inversion if more than 80% of the sequence aligns.

We assess either the ability to predict the presence of an SV or its genotype. For the *presence* evaluation, both heterozygous and homozygous alternate SVs are compared jointly using the approach described above. To compute genotype-level metrics, the heterozygous and homozygous SVs are compared separately. Before splitting the variants by genotype, consecutive heterozygous variants are first stitched together if located at less that 20 bp from each other. Pairs of heterozygous variants with reciprocal overlap of at least 80% are also merged into a homozygous variant before splitting variants by genotype.

Other SV genotypers

BayesTyper (v1.5 beta 62888d6)

Where not specified otherwise BayesTyper was run as follows. Raw reads were mapped to the reference genome using `bwa mem` (0.7.17). GATK[34] (3.8) and Platypus[35] (0.8.1.1) were run on the mapped reads to call SNVs and short indels (<50bp) needed by BayesTyper for correct genotyping. The VCFs with these variants were then normalised using `bcftools norm` (1.9) and combined with the SVs across samples using `bayesTyperTools combine` to produce the input candidate set. k-mers in the raw reads were counted using `kmc` (3.1.1) with a k-mer size of 55. A Bloom filter was constructed from these k-mers using `bayesTyperTools makeBloom`. Finally, variants were clustered and genotyped using `bayestyper cluster` and `bayestyper genotype`, respectively, with default parameters except `--min-genotype-posterior 0`. Non-PASS variants were filtered prior to evaluation using `bcftools filter`.

Delly (v0.7.9)

The `delly call` command was run on the reads mapped by `bwa mem`, the reference genome FASTA file and the VCF containing the SVs to genotype in their explicit representation.

SVTyper (v0.7.0)

The VCF containing deletions was converted to symbolic representation and passed to `svtyper` with the reads mapped by `bwa mem`. The output VCF was converted back to explicit representation using `bayesTyperTools convertAllele` to facilitate variant normalization before evaluation.

SMRT-SV2 (v2.0.0 Feb 21 2019 commit adb13f2)

SMRT-SV2 was run with the “30x-4” model and min-call-depth 8 cutoff. It was run only on VCFs created by SMRT-SV, for which the required contig BAMs were available. The Illumina BAMs used where the same as the other methods described above. The output VCF was converted back to explicit representation to facilitate variant normalization later.

Simulation experiment

We simulated a synthetic genome with 1000 insertions, deletions and inversions. Each variant was separated from the next by a buffer region of 500 bp following the final variable base. The sizes of deletions and insertions followed the distribution of real SV sizes from the HGSVC catalog. We used the same size distribution as deletions for inversions. A VCF file was produced for three simulated samples with genotypes chosen uniformly between homozygous reference, heterozygous, and homozygous alternate.

We created another VCF file containing errors in the SV breakpoint locations. One or both breakpoints of deletions and inversions were shifted between 1 and 10 bp. The locations and sequences of insertions were also modified, either shifting the variants or shortening them at the flanks, again by up to 10 bp.

Paired-end reads were simulated using `vg sim` on the graph that contained the true SVs. Different read depths were tested: 1x, 3x, 7x, 10x, 13x, 20x. We used real Illumina reads from NA12878 provided by the Genome in a Bottle consortium to model base qualities and sequencing errors.

The different methods were tested using either the true VCF or the VCF that contained errors. For vg, a graph was constructed from the VCF file, indexed, then used to map simulated reads and call variants using toil-vg (see toil-vg). BayesTyper was run directly on the simulated reads and using an input VCF with SVs only. In order to run the other methods, reads were mapped to the linear reference sequence using `bwa mem` and sorted using `samtools`. For Delly, insertions and deletions were first genotyped together using these mapped reads and the `delly call` command. Inversions were genotyped separately using a VCF that was formatted according to Delly’s preference. SVTyper was run on the mapped reads and a VCF that was converted to

symbolic variant representation. All commands used for this analysis are available at github.com/vgteam/sv-genotyping-paper.

The genotypes called in each experiment (genotyping method/VCF with or without errors/sequencing depth) were compared to the true SV genotypes to compute the precision, recall and F1 score (see toil-vg sveval).

Breakpoint fine-tuning using graph augmentation

vg can call variants after augmenting the graph with the read alignments to discover new variants (see toil-vg call). We tested if this approach could fine-tune the breakpoint location of SVs in the graph. We started with the graph that contained approximate SVs (1-10 bp errors in breakpoint location) and 20x simulated reads from the simulation experiment (see Simulation experiment). The variants called after graph augmentation were compared with the true SVs and considered fine-tuned if the breakpoints matched exactly.

HGSVC Analysis

Phased VCFs were obtained for the three Human Genome Structural Variation Consortium (HGSVC) samples from Chaisson et al.[19] and combined with `bcftools merge`. A variation graph was created and indexed using the combined VCF and the HS38D1 reference with alt loci excluded. The phasing information was used to construct a GBWT index, from which the two haploid sequences from HG00514 were extracted. Illumina read pairs with 30x coverage were simulated from these sequences using vg, with an error model learned from real reads from the same sample. Still, these reads reflect the idealized situation where the breakpoints of the SVs being genotyped are exactly known a priori. The reads were mapped to the graph and the mappings used to genotype the SVs in the graph, which were finally compared back to the HG00514 genotypes from the HGSVC VCF. The process was repeated with the same reads on the linear reference, using bwa-mem for mapping and Delly, SVTyper and BayesTyper for SV genotyping.

Illumina HiSeq 2500 paired end reads were downloaded from the EBI's ENA FTP site for the three samples, using Run Accessions ERR903030, ERR895347 and ERR894724 for HG00514, HG00733 and NA19240, respectively. The graph and linear mapping and genotyping pipelines were run exactly as for the simulation, and the comparison results were aggregated across the three samples. For BayesTyper the 3 samples were genotyped jointly.

GIAB Analysis

Version 0.6 of the Genome In A Bottle (GIAB) SV VCF for the Ashkenazim son (HG002) was obtained from the NCBI FTP site. Illumina reads downsampled to 50x coverage obtained as described in Garrison et al.[14], were used to run the vg and linear SV genotyping pipelines described above though with GRCh37 instead of 38. For BayesTyper the input variant set was

created by combining the GIAB SVs with SNV and indels from the same study. Variants without a determined genotype (14649 out of 74012), which correspond to putative technical artifacts and parental calls not present in HG002, were considered “false positives” as a proxy measure for precision.

SMRT-SV2 Comparison (CHMPD and SVPOP)

The SMRT-SV2 genotyper can only be used to genotype VCFs that were created by SMRT-SV2, and therefore could not be run on our simulated, HGSVC or GIAB data. The authors shared their training and evaluation set, a pseudodiploid sample constructed from combining the haploid CHM1 and CHM13 samples (CHMPD), along with a negative control (NA19240). The high quality of the CHM assemblies makes this set an attractive alternative to using simulated reads. We used this two-sample pseudodiploid VCF along with the 30X read set to construct, map and genotype with vg, and also ran SMRT-SV2 genotyper with the “30x-4” model and min-call-depth 8 cutoff, and compared the two back to the original VCF.

In an effort to extend this comparison to a more realistic setting, we reran the three HGSVC samples against the SMRT-SV2 discovery VCF (SVPOP, which contains them in addition to 12 other samples) published by Audano et al.[4] using vg and SMRT-SV2 Genotyper. The discovery VCF does not contain genotypes so we did not distinguish between heterozygous and homozygous genotypes, looking at only the presence or absence of an alt allele for each variant.

SMRT-SV2 produces some explicit *no-calls* predictions when the read coverage is too low to produce accurate genotypes. These no-calls are considered homozygous reference in the main accuracy evaluation. We also explored the performance of vg and SMRT-SV2 in different sets of regions (Figure S11 and Table S5):

1. Non-repeat regions, i.e. excluding segmental duplications and tandem repeats.
2. Repeat regions defined as segmental duplications and tandem repeats.
3. Regions where SMRT-SV2 could call variants.
4. Regions where SMRT-SV2 produced no-calls.

Yeast graph analysis

For the analysis of graphs from de novo assemblies, we utilized publicly available PacBio-derived assemblies and Illumina short read sequencing datasets for 12 yeast strains from two related clades (Table 1) [24]. Five strains were selected (two from different subclades of each clade plus the reference *S.c. S288C*): *S.c. SK1*, *S.c. YPS128*, *S.p. CBS432*, *S.p. UFRJ50816*, and *S.c. S288C*. Two different genome graphs were constructed from the assemblies of the five selected strains. In the following, we describe the steps for the construction of both graphs and the calling of variants. More details and the precise commands used in our analyses can be found at github.com/vgteam/sv-genotyping-paper.

Table 1: 12 yeast strains from two related clades were used in our analysis. Five strains were selected to be included in the graphs while the remaining seven were used for variant calling only.

Strain	Clade	Included in graph
S288C	S. cerevisiae	✓
SK1	S. cerevisiae	✓
YPS128	S. cerevisiae	✓
UWOPS034614	S. cerevisiae	
Y12	S. cerevisiae	
DBVPG6765	S. cerevisiae	
DBVPG6044	S. cerevisiae	
CBS432	S. paradoxus	✓
UFRJ50816	S. paradoxus	✓
N44	S. paradoxus	
UWOPS919171	S. paradoxus	
YPS138	S. paradoxus	

Construction of the *VCF graph*

For the first graph (called the *VCF graph* throughout the paper), the default vg graph construction method was applied. It requires a linear reference genome and a VCF file of variants on that reference to build the graph. As reference genome, the PacBio assembly of the S.c. S288C strain was chosen because this strain was used for the S. cerevisiae genome reference assembly. To obtain variants three methods for SV detection from genome assemblies were combined: Assemblytics [25] (commit df5361f), AsmVar (commit 5abd91a) [26] and paftools (version 2.14-r883) [27]. All three methods were run to detect SVs between the PacBio assembly of reference strain S.c. S288C and the PacBio assemblies of each of the four other selected yeast strains. The union of variants detected by the three methods was produced (using bedtools [36]) and variants with a reciprocal overlap of at least 50% were combined to avoid duplication in the union set. These union sets of variants for each of the four selected (and non-reference) strains were merged and another deduplication step was applied to combine variants with a reciprocal overlap of at

least 90%. The resulting total set of variants in VCF format and the linear reference genome were used to build the *VCF graph* with `vg construct`.

Construction of the *cactus graph*

For the second graph (called the *cactus graph* throughout the paper), an alternative graph construction methods directly from de novo genome assemblies was applied. First, the repeat-masked PacBio-assemblies of the five selected strains were aligned with our Cactus tool [23]. Cactus requires a phylogenetic tree of the strains which was estimated using Mash (version 2.1) [37] and PHYLIP (version 3.695) [38]. Subsequently, the output file in HAL format was converted to a variant graph with hal2vg (https://github.com/ComparativeGenomicsToolkit/hal2vg).

Calling and genotyping of SVs

Prior to variant calling, the Illumina short reads of all 12 yeast strains were mapped to both graphs using `vg map`. The fractions of reads mapped with specific properties were measured using `vg view` and the JSON processor `jq`. Then, `toil-vg call` (commit be8b6da) was used to analyze the mapped reads of each of the 11 non-reference strains and to call variants. Thus, a separate variant callset was obtained for each of the strains and both graphs. To evaluate the callsets, a sample graph (i.e. a graph representation of the callset) was generated for each callset using `vg construct` and `vg mod` on the reference assembly *S.c. S288C* and the callset. Subsequently, short reads from the respective strains were mapped to each sample graph using `vg map`. The resulting alignments were analyzed with `vg view` and `jq`.

Supplementary Material

Supplementary Tables

Table S1: Genotyping evaluation on the HGSVC dataset. Precision, recall and F1 score for the call set with the best F1 score. The numbers in parentheses corresponds to the results in non-repeat regions.

Experiment	Method	Type	Precision	Recall	F1
Simulated reads	vg	INS	0.795 (0.885)	0.796 (0.883)	0.795 (0.884)
		DEL	0.869 (0.971)	0.771 (0.92)	0.817 (0.945)
	BayesTyper	INS	0.91 (0.935)	0.835 (0.9)	0.871 (0.917)
		DEL	0.898 (0.981)	0.806 (0.929)	0.849 (0.954)
	SVTyper	DEL	0.809 (0.876)	0.328 (0.754)	0.467 (0.81)
	Delly	INS	0.767 (0.866)	0.093 (0.225)	0.166 (0.358)
		DEL	0.696 (0.903)	0.707 (0.846)	0.701 (0.874)
Real reads	vg	INS	0.431 (0.683)	0.541 (0.726)	0.48 (0.704)
		DEL	0.65 (0.886)	0.519 (0.708)	0.577 (0.787)
	BayesTyper	INS	0.601 (0.747)	0.254 (0.433)	0.357 (0.549)
		DEL	0.627 (0.91)	0.325 (0.381)	0.428 (0.537)
	SVTyper	DEL	0.661 (0.733)	0.236 (0.551)	0.348 (0.629)
	Delly	INS	0.516 (0.621)	0.068 (0.176)	0.12 (0.275)

Experiment	Method	Type	Precision	Recall	F1
		DEL	0.55 (0.838)	0.445 (0.547)	0.492 (0.662)

Table S2: Genotyping evaluation on the Genome in a Bottle dataset. Precision, recall and F1 score for the call set with the best F1 score. The numbers in parentheses corresponds to the results in non-repeat regions.

Method	Type	Precision	Recall	F1
vg	INS	0.658 (0.774)	0.646 (0.735)	0.652 (0.754)
	DEL	0.68 (0.768)	0.643 (0.735)	0.661 (0.751)
BayesTyper	INS	0.776 (0.879)	0.286 (0.379)	0.418 (0.53)
	DEL	0.808 (0.886)	0.512 (0.696)	0.627 (0.779)
SVTyper	DEL	0.742 (0.818)	0.342 (0.496)	0.468 (0.618)
Delly	INS	0.822 (0.894)	0.177 (0.268)	0.291 (0.412)
	DEL	0.722 (0.822)	0.645 (0.768)	0.681 (0.794)

Table S3: Genotyping evaluation on the pseudo-diploid genome built from CHM cell lines in Audano et al.[4].

Method	Region	Type	Precision	Recall	F1
vg	all	INS	0.665	0.661	0.663
		DEL	0.688	0.500	0.579
	non-repeat	INS	0.806	0.784	0.795
		DEL	0.869	0.762	0.812
SMRT-SV2	all	INS	0.757	0.536	0.628
		DEL	0.848	0.630	0.723
	non-repeat	INS	0.880	0.680	0.767
		DEL	0.971	0.824	0.891

Table S4: Calling evaluation on the SVPOP dataset. Combined results for the HG00514, HG00733 and NA19240 individuals, 3 of the 15 individuals used to generate the high-quality SV catalog in Audano et al.[4].

Method	Region	Type	TP	FP	FN	Precision	Recall	F1
vg	all	INS	25838	22042	15772	0.540	0.621	0.577
		DEL	14545	6824	15425	0.681	0.485	0.567
		INV	27	26	173	0.509	0.135	0.213
	non-repeat	INS	8051	3258	1817	0.712	0.816	0.760
		DEL	3769	623	818	0.858	0.822	0.840
		INV	19	12	75	0.613	0.202	0.304
SMRT-SV2	all	INS	16270	26031	25340	0.385	0.391	0.388
		DEL	11793	10106	18177	0.539	0.393	0.455
	non-repeat	INS	4483	4659	5385	0.490	0.454	0.472
		DEL	2928	930	1659	0.759	0.638	0.693

Table S5: Calling evaluation on the SVPOP dataset in different sets of regions for the HG5014 individual.

Method	Region	Type	TP	FP	FN	Precision	Recall	F1
vg	all	INS	8618	7237	5416	0.546	0.614	0.578
		DEL	4762	2048	5145	0.696	0.481	0.569
		INV	11	8	54	0.579	0.169	0.262
	repeat	INS	6176	6923	4678	0.475	0.569	0.518
		DEL	2428	1701	4542	0.584	0.348	0.436
		INV	1	1	6	0.500	0.143	0.222
	non-repeat	INS	2677	987	514	0.731	0.839	0.781
		DEL	1180	176	321	0.869	0.786	0.825
		INV	7	4	20	0.636	0.259	0.368
	called in SMRT-SV	INS	3410	3789	2108	0.478	0.618	0.539
		DEL	2544	1092	1518	0.699	0.626	0.661
		INV	8	8	52	0.500	0.133	0.210
	not called in SMRT-SV	INS	4838	542	3678	0.899	0.568	0.696
		DEL	2034	26	3723	0.987	0.353	0.520
SMRT-SV2	all	INS	5245	8563	8789	0.394	0.374	0.384
		DEL	3741	3382	6166	0.533	0.378	0.442
	repeat	INS	3848	7125	7006	0.368	0.354	0.361
		DEL	1990	2832	4980	0.426	0.286	0.342

Method	Region	Type	TP	FP	FN	Precision	Recall	F1
	non-repeat	INS	1396	1468	1795	0.493	0.438	0.464
		DEL	901	308	600	0.745	0.600	0.665
	called in SMRT-SV	INS	4343	5595	1175	0.445	0.787	0.569
		DEL	3227	2451	835	0.573	0.794	0.666
	not called in SMRT-SV	INS	116	109	8400	0.551	0.014	0.026
		DEL	206	16	5551	0.911	0.036	0.069

Table S6: Breakpoint fine-tuning using graph augmentation from the read alignment. For deletions and inversions, either one or both breakpoints were shifted to introduce errors in the input VCF. For insertions, the insertion location and sequence contained errors. In all cases, the errors affected 1-10 bp.

SV type	Error type	Breakpoint	Variant	Proportion	Mean size (bp)	Mean error (bp)
DEL	one end	incorrect	220	0.219	422.655	6.095
		fine-tuned	784	0.781	670.518	5.430
	both ends	incorrect	811	0.814	826.070	6.275
		fine-tuned	185	0.186	586.676	2.232
INS	location/ seq	incorrect	123	0.062	428.724	6.667
		fine-tuned	1877	0.938	440.043	6.439
INV	one end	incorrect	868	0.835	762.673	5.161
		fine-tuned	172	0.165	130.244	5.884
	both ends	incorrect	950	0.992	556.274	5.624
		fine-tuned	8	0.008	200.000	1.375

Supplementary Figures

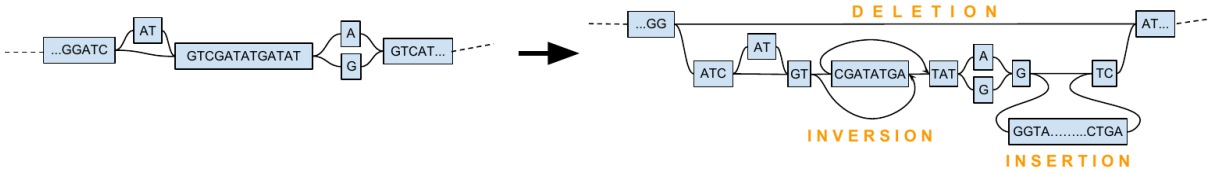


Figure S1: **Adding large insertions, deletions and inversions in a variation graph.**

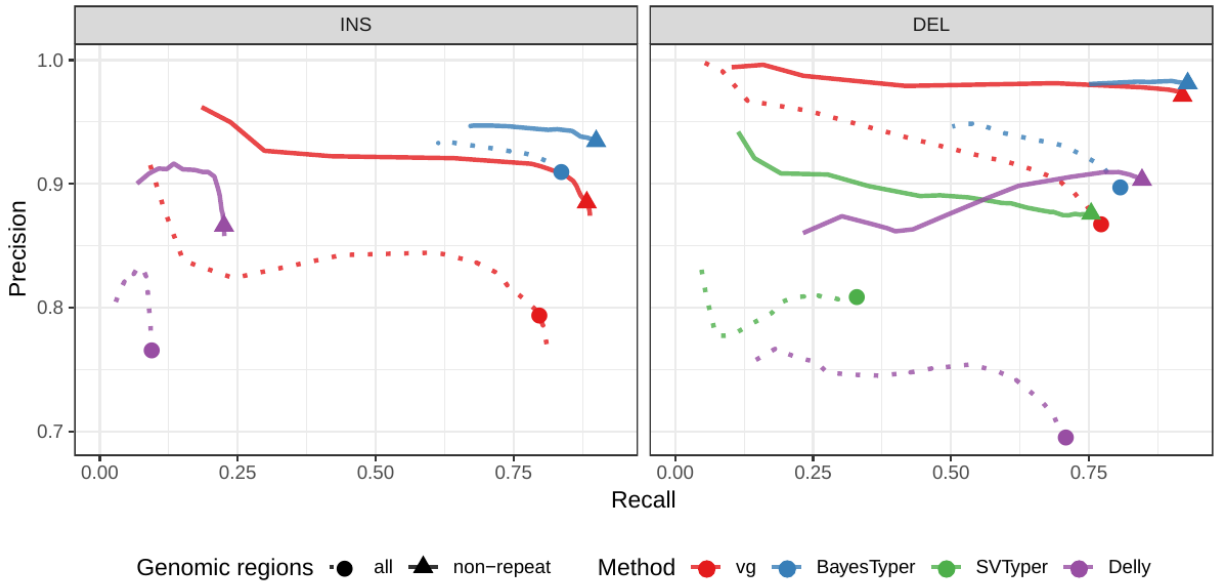


Figure S2: **Genotyping evaluation on the HGSVC dataset using simulated reads.**

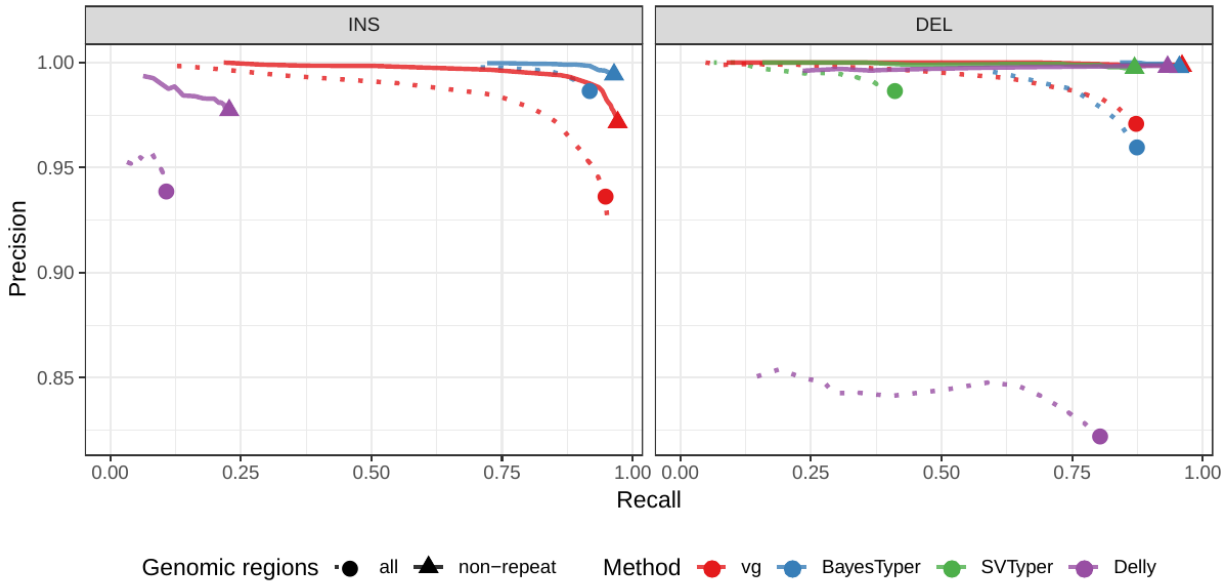


Figure S3: **Calling evaluation on the HGSVC dataset using simulated reads.**

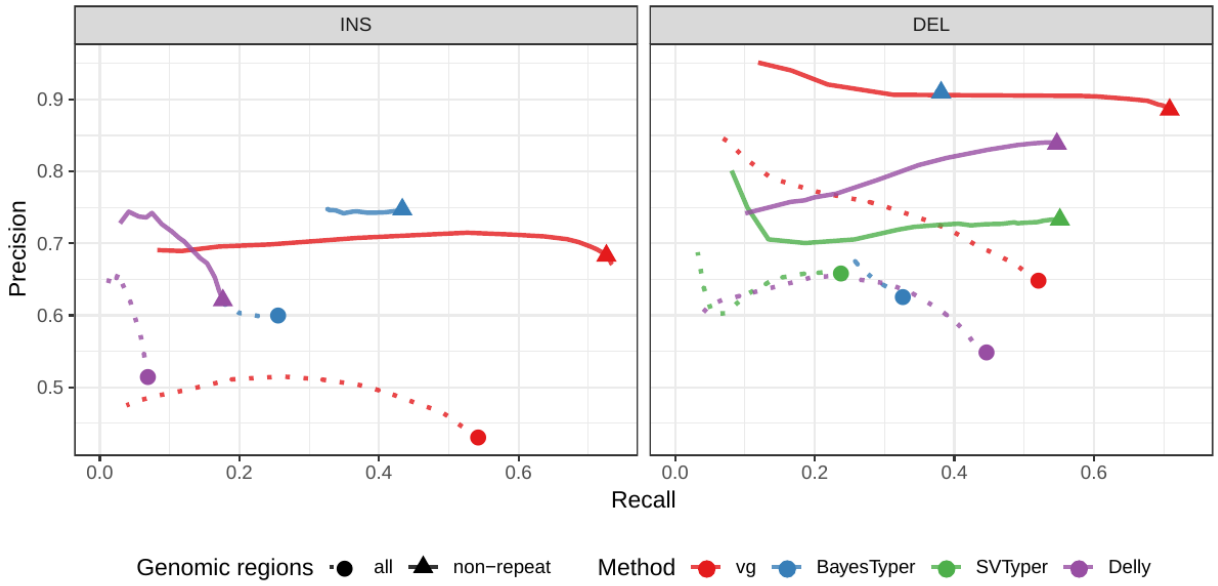


Figure S4: **Genotyping evaluation on the HGSVC dataset using real reads.** Combined results across the HG00514, HG00733 and NA19240.

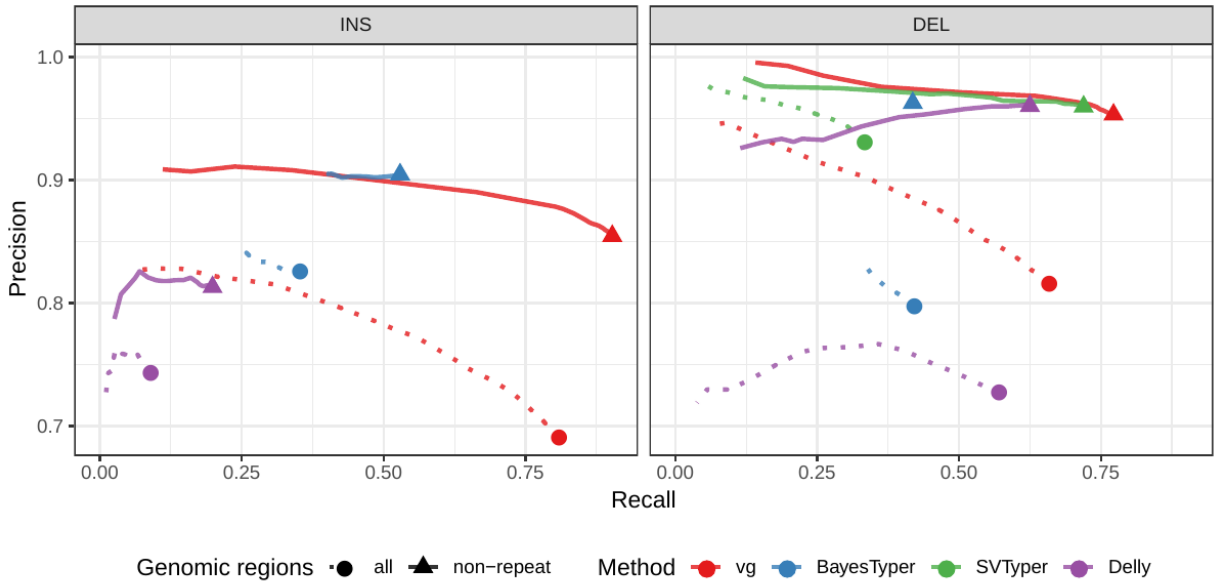


Figure S5: **Calling evaluation on the HGSVC dataset using real reads.** Combined results across the HG00514, HG00733 and NA19240.

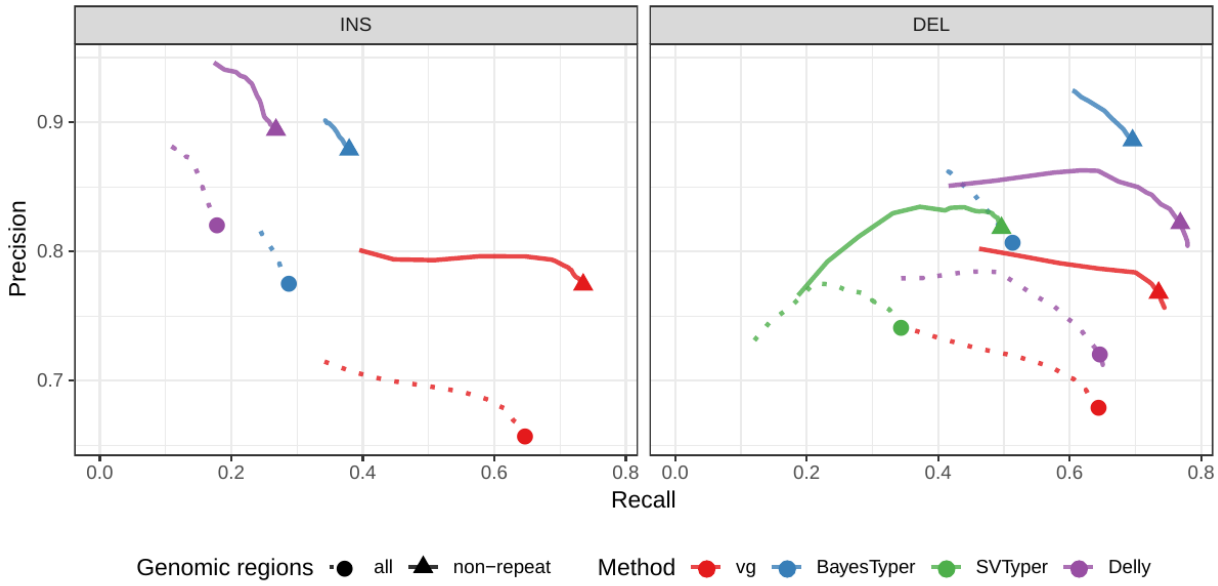


Figure S6: **Genotyping evaluation on the Genome in a Bottle dataset.** Predicted genotypes on HG002 were compared to the high-quality SVs from this same individual.

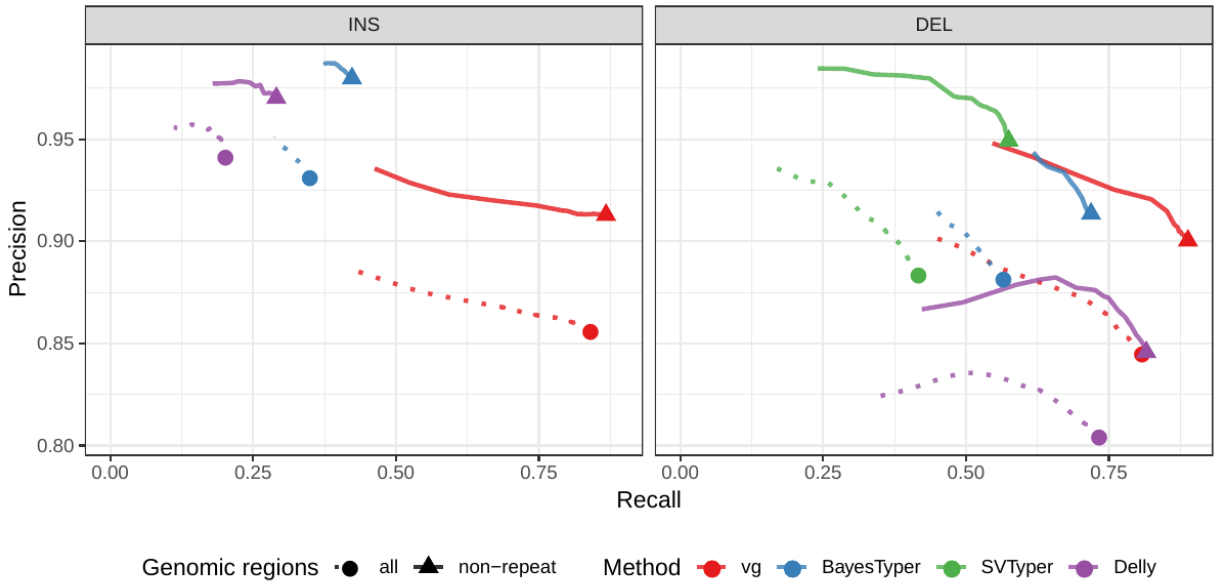


Figure S7: **Calling evaluation on the Genome in a Bottle dataset.** Calls on HG002 were compared to the high-quality SVs from this same individual.

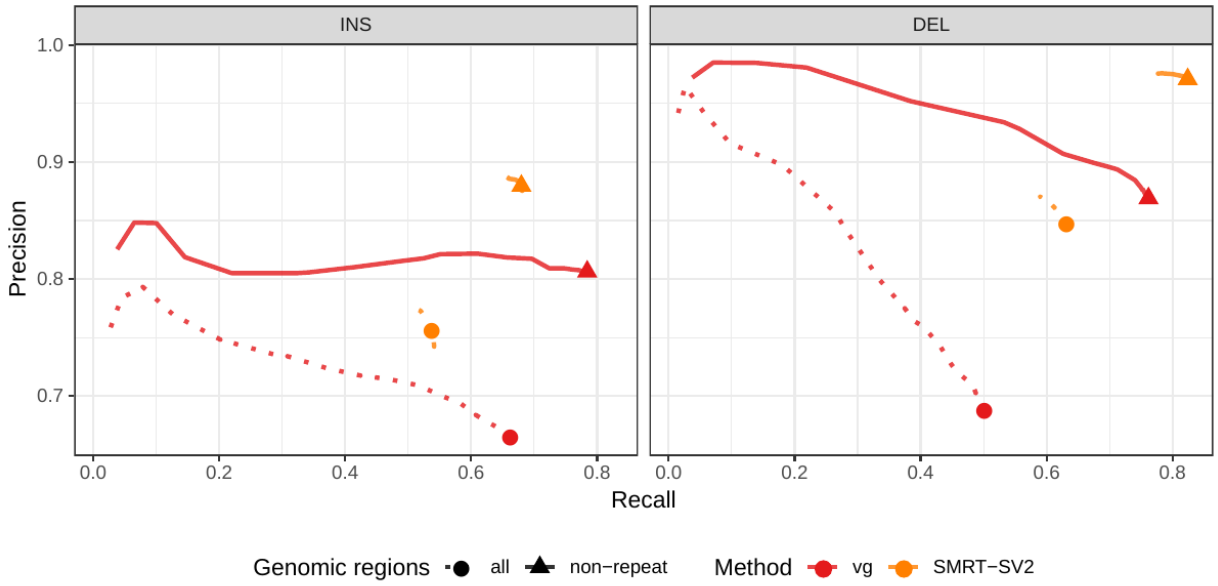


Figure S8: **Genotyping evaluation on the CHM pseudo-diploid dataset.** The pseudo-diploid genome was built from CHM cell lines and used to train SMRT-SV2 in Audano et al.[4]

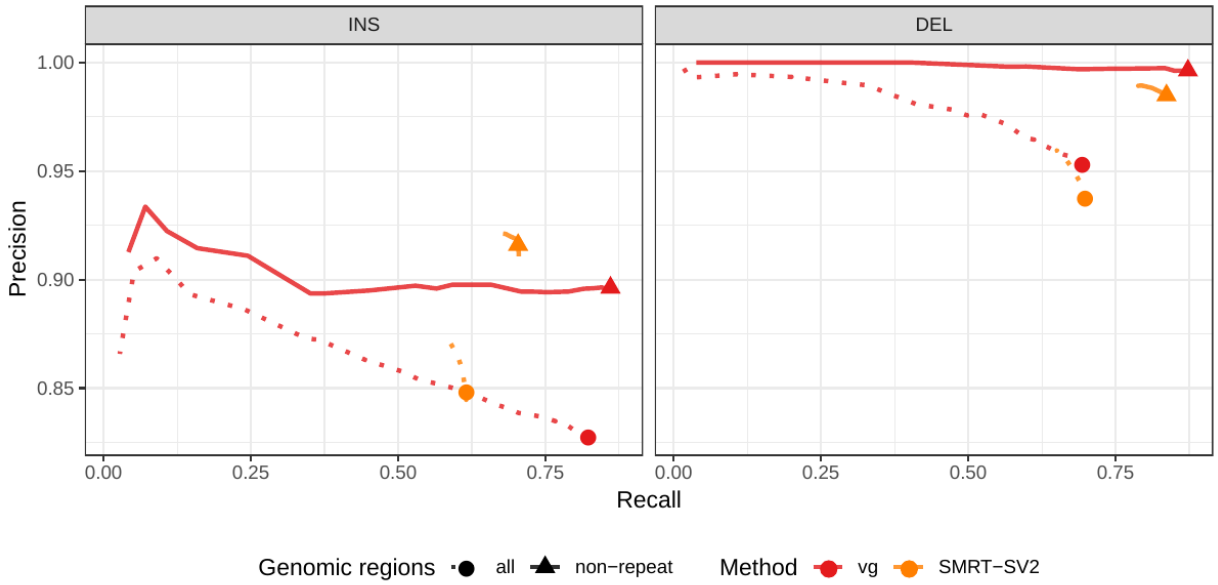


Figure S9: **Calling evaluation on the CHM pseudo-diploid dataset.** The pseudo-diploid genome was built from CHM cell lines and used to train SMRT-SV2 in Audano et al.[4]

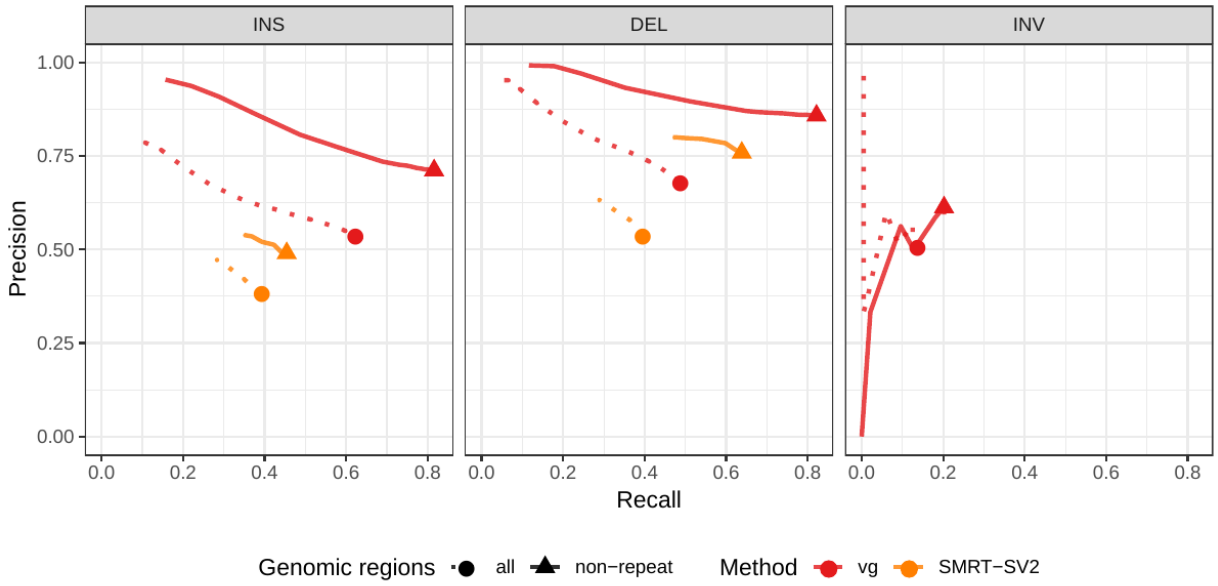


Figure S10: **Calling evaluation on the SVPOP dataset.** Combined results across the HG00514, HG00733 and NA19240.

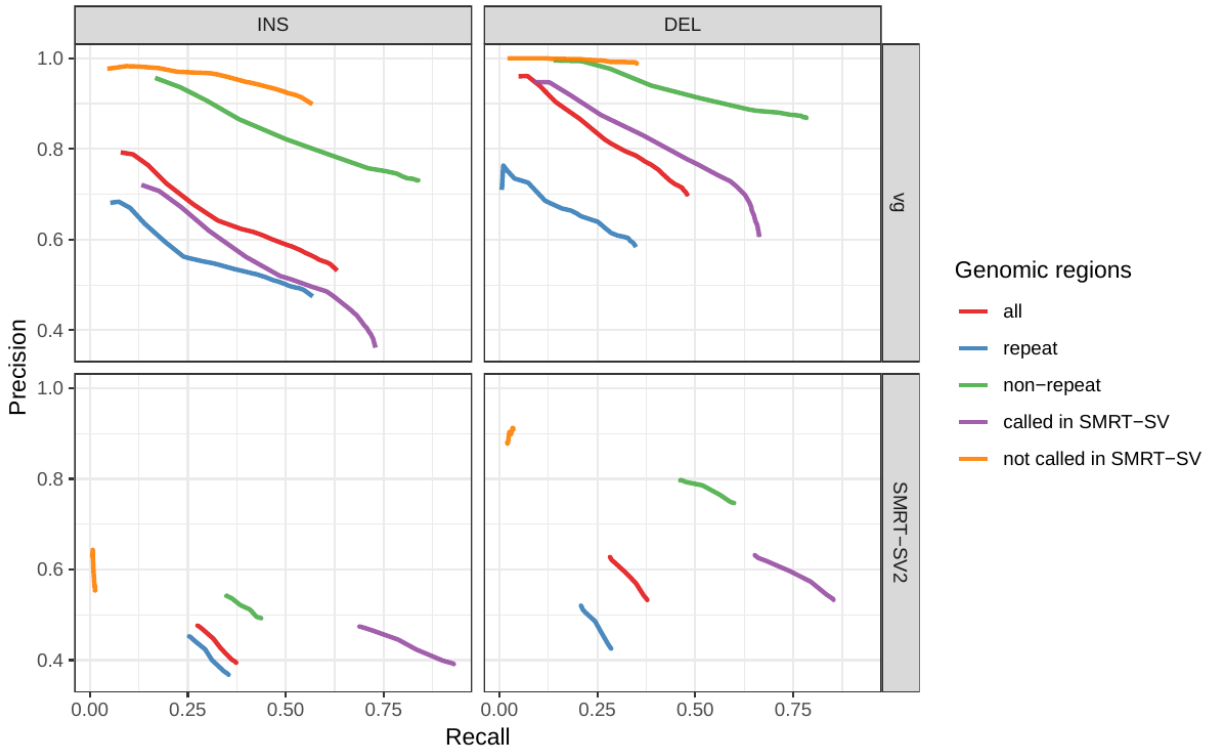


Figure S11: **Evaluation across different sets of regions in HG00514 (SVPOP dataset)**. Calling evaluation.

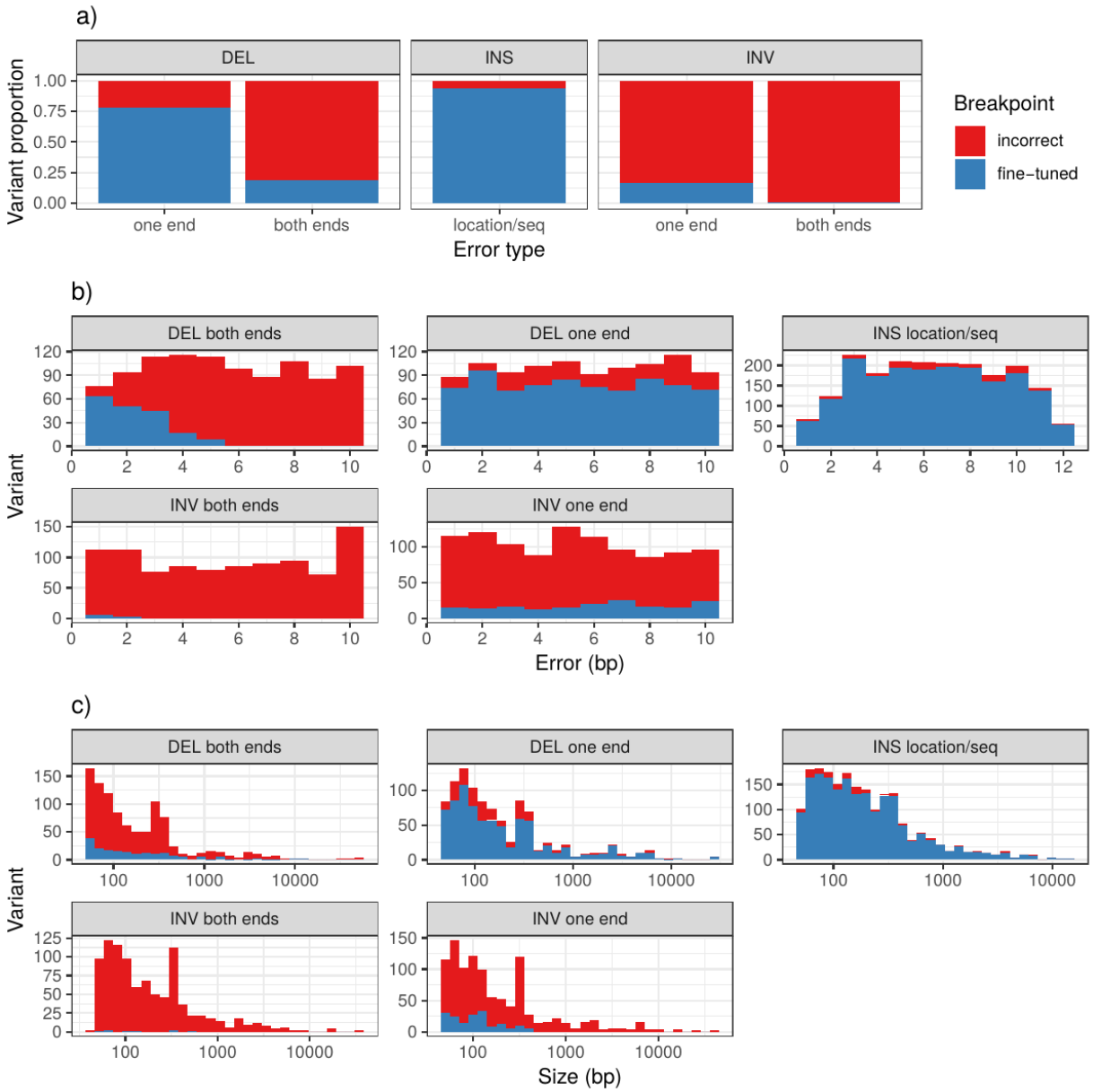


Figure S12: **Breakpoint fine-tuning using augmentation through "vg call".** For deletions and inversions, either one or both breakpoints were shifted to introduce errors in the input VCF. For insertions, the insertion location and sequence contained errors. a) Proportion of variant for which breakpoints could be fine-tuned. b) Distribution of the amount of errors that could be corrected or not. c) Distribution of the size of the variants whose breakpoints could be fine-tuned or not.

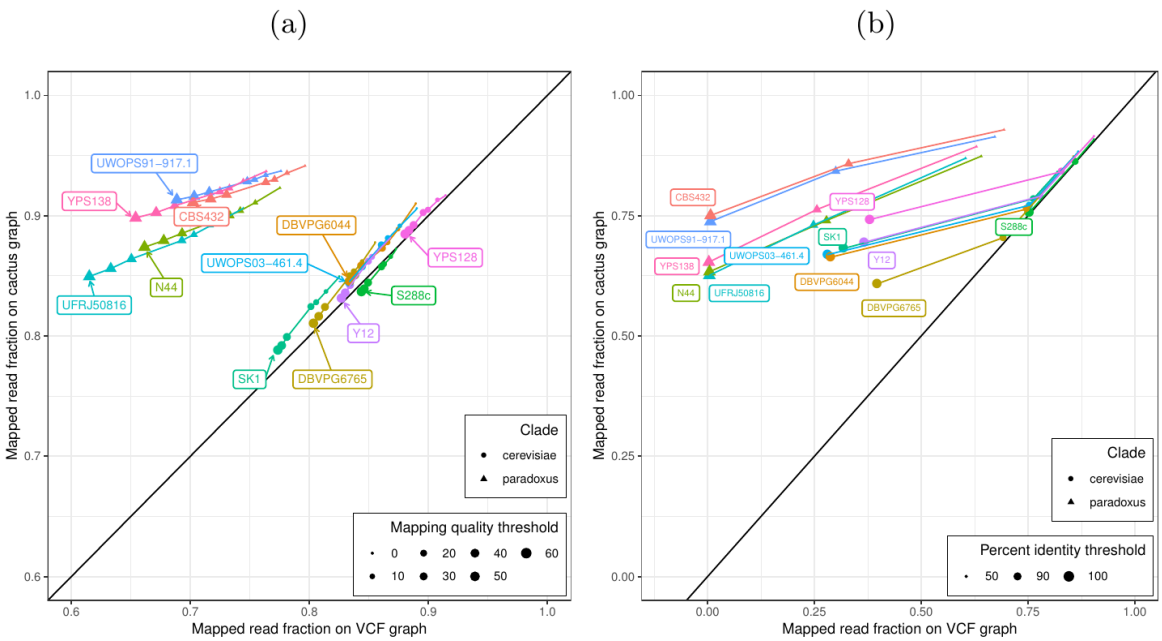


Figure S13: **Mapping comparison on graphs of all 12 strains.** Short reads from all 12 yeast strains were aligned to both graphs. The fraction of reads mapped to the *cactus graph* (y-axis) and the *VCF graph* (x-axis) are compared. a) Stratified by mapping quality threshold. b) Stratified by percent identity threshold. Colors and shapes represent the 12 strains and two clades, respectively.

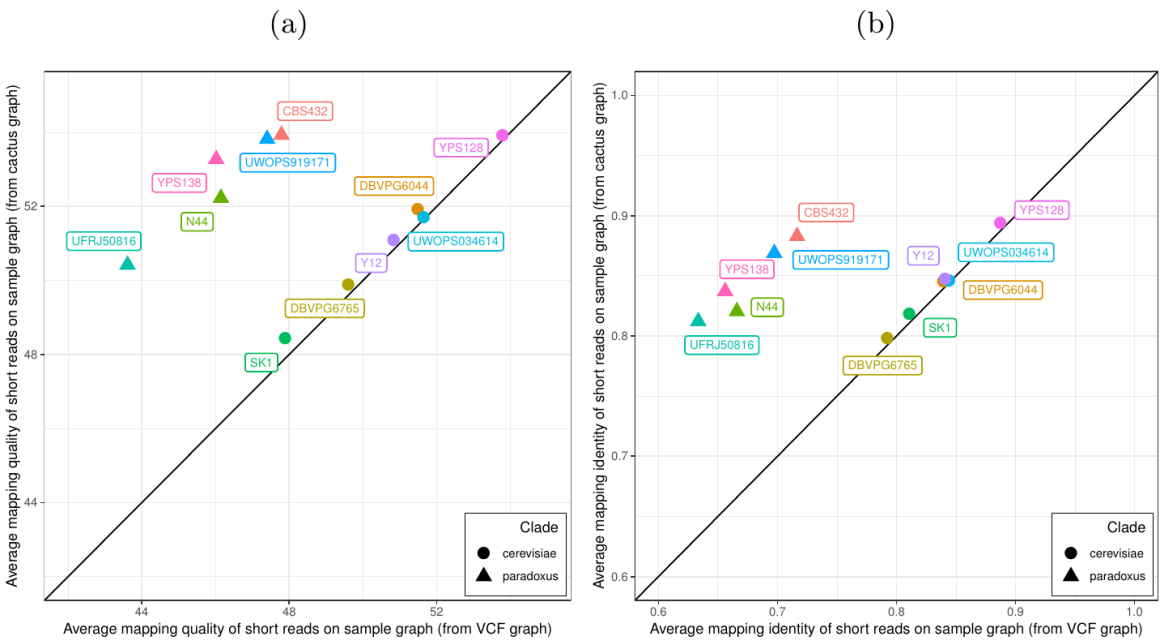


Figure S14: **SV genotyping comparison on graphs of all 12 strains.** Short reads from all 11 non-reference yeast strains were used to genotype SVs contained in both graphs. Subsequently, sample graphs were generated from the resulting SV callsets. The short reads were again aligned to the sample graphs and the quality of the alignments was used to ascertain genotyping performance. a) Average mapping quality of short reads aligned to the sample graphs derived from *cactus graph* (y-axis) and *VCF graph* (x-axis). b) Average mapping identity of short reads aligned to the sample graphs derived from *cactus graph* (y-axis) and *VCF graph* (x-axis). Colors and shapes represent the 11 non-reference strains and two clades, respectively.

Supplementary Information

Variation graph and structural variation

A variation graph encodes DNA sequence in its nodes. Such graphs are bidirected, in that we distinguish between edges incident on the starts of nodes from those incident on their ends. A path in such a graph is an ordered list of nodes where each is associated with an orientation. If a path walks from, for example, node A in the forward orientation to node B in the reverse orientation, then an edge must exist from the end of node A to the end of node B. Concatenating the sequences on each node in the path, taking the reverse complement when the node is visited in reverse orientation, produces a DNA sequence. Accordingly, variation graphs are constructed so as to encode haplotype sequences as walks through the graph. Variation between sequences shows up as bubbles in the graph [22]. Figure S1 shows how a graph with a SNP and an indel can be extended to contain more complex SVs.

Breakpoint fine-tuning

In addition to genotyping, vg can use an augmentation step to modify the graph based on the read alignment and discover novel variants. On the simulated SVs from Figure 1b, this approach was able to correct many of the 1-10 bp breakpoint errors that were added to the input VCF. The breakpoints were accurately fine-tuned for 93.8% of the insertions (Figure S12a and Table S6). For deletions, 78.1% of the variants were corrected when only one breakpoint had an error. In situations where both breakpoints of the deletions were incorrect, only 18.6% were corrected through graph augmentation, and only when the amount of error was small (Figure S12b). The breakpoints of less than 20% of the inversions could be corrected. Across all SV types, the size of the variant didn't affect the ability to fine-tune the breakpoints through graph augmentation (Figure S12c).

Mapping identity in genome graphs from 12 yeast strains

Generally, more reads were mapped to the *cactus graph* than to the *VCF graph*. We observed a similar trend when exploring the mapping identity of the short reads on the graphs (see Figure 5b). For strains in the *S. paradoxus* clade, the *cactus graph* resulted in substantially more mappings with high percent identity than the *VCF graph*. With strains in the *S. cerevisiae* clade, the difference was smaller, at least for a percent identity threshold up to 90%. When comparing read fractions with perfect identity (i.e. percent identity threshold = 100%), the *cactus graph* clearly outperforms the *VCF graph* on 11 out of 12 samples, the exception again being the reference strain *S.c. S288C*. One reason behind this is that the VCF graph only contains SVs but no SNPs or Indels. The cactus graph which is constructed directly from de novo assemblies consequently captures the genetic makeup of each strain more comprehensively and accurately.

References

1. **Phenotypic impact of genomic structural variation: insights from and for human disease**
Joachim Weischenfeldt, Orsolya Symmons, François Spitz, Jan O. Korbel
Nature Reviews Genetics (2013-02) https://doi.org/f4nhxh
DOI: 10.1038/nrg3373 · PMID: 23329113

2. **SpeedSeq: ultra-fast personal genome analysis and interpretation**
Colby Chiang, Ryan M Layer, Gregory G Faust, Michael R Lindberg, David B Rose, Erik P Garrison, Gabor T Marth, Aaron R Quinlan, Ira M Hall
Nature Methods (2015-08-10) https://doi.org/gcpgfh
DOI: 10.1038/nmeth.3505 · PMID: 26258291 · PMCID: PMC4589466

3. **DELLY: structural variant discovery by integrated paired-end and split-read analysis**
T. Rausch, T. Zichner, A. Schlattl, A. M. Stutz, V. Benes, J. O. Korbel
Bioinformatics (2012-09-07) https://doi.org/f38r2c
DOI: 10.1093/bioinformatics/bts378 · PMID: 22962449 · PMCID: PMC3436805

4. **Characterizing the Major Structural Variant Alleles of the Human Genome**
Peter A. Audano, Arvis Sulovari, Tina A. Graves-Lindsay, Stuart Cantsilieris, Melanie Sorensen, AnneMarie E. Welch, Max L. Dougherty, Bradley J. Nelson, Ankeeta Shah, Susan K. Dutcher, … Evan E. Eichler
Cell (2019-01) https://doi.org/gfthvz
DOI: 10.1016/j.cell.2018.12.019 · PMID: 30661756

5. **An integrated map of structural variation in 2,504 human genomes**
Peter H. SudmantTobias Rausch, Eugene J. Gardner, Robert E. Handsaker, Alexej Abyzov, John Huddleston, Yan Zhang, Kai Ye, Goo Jun, … Jan O. Korbel
Nature (2015-10) https://doi.org/73c
DOI: 10.1038/nature15394 · PMID: 26432246 · PMCID: PMC4617611

6. **Whole-genome sequence variation, population structure and demographic history of the Dutch population**
Laurent C FrancioliAndroniki Menelaou, Sara L Pulit, Freerk van Dijk, Pier Francesco Palamara, Clara C Elbers, Pieter BT Neerincx, Kai Ye, Victor Guryev, … Cisca Wijmenga
Nature Genetics (2014-06-29) https://doi.org/f6bxm8
DOI: 10.1038/ng.3021 · PMID: 24974849

7. **Resolving the complexity of the human genome using single-molecule sequencing**
Mark J. P. Chaisson, John Huddleston, Megan Y. Dennis, Peter H. Sudmant, Maika Malig, Fereydoun Hormozdiari, Francesca Antonacci, Urvashi Surti, Richard Sandstrom, Matthew

Boitano, … Evan E. Eichler
Nature (2014-11-10) https://doi.org/w69
DOI: 10.1038/nature13907 · PMID: 25383537 · PMCID: PMC4317254

8. **Discovery and genotyping of structural variation from long-read haploid genome sequence data**
John Huddleston, Mark J.P. Chaisson, Karyn Meltz Steinberg, Wes Warren, Kendra Hoekzema, David Gordon, Tina A. Graves-Lindsay, Katherine M. Munson, Zev N. Kronenberg, Laura Vives, … Evan E. Eichler
Genome Research (2016-11-28) https://doi.org/f9x79h
DOI: 10.1101/gr.214007.116 · PMID: 27895111 · PMCID: PMC5411763

9. **Mapping and phasing of structural variation in patient genomes using nanopore sequencing**
Mircea Cretu Stancu, Markus J. van Roosmalen, Ivo Renkens, Marleen M. Nieboer, Sjors Middelkamp, Joep de Ligt, Giulia Pregno, Daniela Giachino, Giorgia Mandrile, Jose Espejo Valle-Inclan, … Wigard P. Kloosterman
Nature Communications (2017-11-06) https://doi.org/gftpt9
DOI: 10.1038/s41467-017-01343-4 · PMID: 29109544 · PMCID: PMC5673902

10. **Genome-wide reconstruction of complex structural variants using read clouds**
Noah Spies, Ziming Weng, Alex Bishara, Jennifer McDaniel, David Catoe, Justin M Zook, Marc Salit, Robert B West, Serafim Batzoglou, Arend Sidow
Nature Methods (2017-07-17) https://doi.org/gbnhwk
DOI: 10.1038/nmeth.4366 · PMID: 28714986 · PMCID: PMC5578891

11. **Nanopore sequencing and assembly of a human genome with ultra-long reads**
Miten Jain, Sergey Koren, Karen H Miga, Josh Quick, Arthur C Rand, Thomas A Sasani, John R Tyson, Andrew D Beggs, Alexander T Dilthey, Ian T Fiddes, … Matthew Loose
Nature Biotechnology (2018-01-29) https://doi.org/gczffw
DOI: 10.1038/nbt.4060 · PMID: 29431738 · PMCID: PMC5889714

12. **Phased diploid genome assembly with single-molecule real-time sequencing**
Chen-Shan Chin, Paul Peluso, Fritz J Sedlazeck, Maria Nattestad, Gregory T Concepcion, Alicia Clum, Christopher Dunn, Ronan O'Malley, Rosa Figueroa-Balderas, Abraham Morales-Cruz, … Michael C Schatz
Nature Methods (2016-10-17) https://doi.org/f9fv4w
DOI: 10.1038/nmeth.4035 · PMID: 27749838 · PMCID: PMC5503144

13. **Genome graphs and the evolution of genome inference**
Benedict Paten, Adam M. Novak, Jordan M. Eizenga, Erik Garrison
Genome Research (2017-03-30) https://doi.org/f95nhd
DOI: 10.1101/gr.214155.116 · PMID: 28360232 · PMCID: PMC5411762

14. **Variation graph toolkit improves read mapping by representing genetic variation in the reference**
Erik Garrison, Jouni Sirén, Adam M Novak, Glenn Hickey, Jordan M Eizenga, Eric T Dawson, William Jones, Shilpa Garg, Charles Markello, Michael F Lin, … Richard Durbin
Nature Biotechnology (2018-08-20) https://doi.org/gd2zqs
DOI: 10.1038/nbt.4227 · PMID: 30125266 · PMCID: PMC6126949

15. **Genome Graphs**
Adam M Novak, Glenn Hickey, Erik Garrison, Sean Blum, Abram Connelly, Alexander Dilthey, Jordan Eizenga, M. A. Saleh Elmohamed, Sally Guthrie, André Kahles, … Benedict Paten
Cold Spring Harbor Laboratory (2017-01-18) https://doi.org/gdcc74
DOI: 10.1101/101378

16. **Fast and accurate genomic analyses using genome graphs**
Goran Rakocevic, Vladimir Semenyuk, Wan-Ping Lee, James Spencer, John Browning, Ivan J. Johnson, Vladan Arsenijevic, Jelena Nadj, Kaushik Ghose, Maria C. Suciu, … Deniz Kural
Nature Genetics (2019-01-14) https://doi.org/gftd46
DOI: 10.1038/s41588-018-0316-4 · PMID: 30643257

17. **Graphtyper enables population-scale genotyping using pangenome graphs**
Hannes P Eggertsson, Hakon Jonsson, Snaedis Kristmundsdottir, Eirikur Hjartarson, Birte Kehr, Gisli Masson, Florian Zink, Kristjan E Hjorleifsson, Aslaug Jonasdottir, Adalbjorg Jonasdottir, … Bjarni V Halldorsson
Nature Genetics (2017-09-25) https://doi.org/gbx7v6
DOI: 10.1038/ng.3964 · PMID: 28945251

18. **Accurate genotyping across variant classes and lengths using variant graphs**
Jonas Andreas SibbesenLasse Maretty, Anders Krogh
Nature Genetics (2018-06-18) https://doi.org/gdndnz
DOI: 10.1038/s41588-018-0145-5 · PMID: 29915429

19. **Multi-platform discovery of haplotype-resolved structural variation in human genomes**
Mark J.P. Chaisson, Ashley D. Sanders, Xuefang Zhao, Ankit Malhotra, David Porubsky, Tobias Rausch, Eugene J. Gardner, Oscar Rodriguez, Li Guo, Ryan L. Collins, … Charles Lee
Cold Spring Harbor Laboratory (2017-09-23) https://doi.org/gftxhc
DOI: 10.1101/193144

20. **Extensive sequencing of seven human genomes to characterize benchmark reference materials**
Justin M. Zook, David Catoe, Jennifer McDaniel, Lindsay Vang, Noah Spies, Arend Sidow, Ziming Weng, Yuling Liu, Christopher E. Mason, Noah Alexander, … Marc Salit

Scientific Data (2016-06-07) https://doi.org/f84nqc
DOI: 10.1038/sdata.2016.25 · PMID: 27271295 · PMCID: PMC4896128

21. **Reproducible integration of multiple sequencing datasets to form high-confidence SNP, indel, and reference calls for five human genome reference materials**
Justin Zook, Jennifer McDaniel, Hemang Parikh, Haynes Heaton, Sean A Irvine, Len Trigg, Rebecca Truty, Cory Y McLean, Francisco M De La Vega, Chunlin Xiao, …
Cold Spring Harbor Laboratory (2018-03-13) https://doi.org/gfwsmj
DOI: 10.1101/281006

22. **Superbubbles, Ultrabubbles, and Cacti**
Benedict Paten, Jordan M. Eizenga, Yohei M. Rosen, Adam M. Novak, Erik Garrison, Glenn Hickey
Journal of Computational Biology (2018-07) https://doi.org/gdw582
DOI: 10.1089/cmb.2017.0251 · PMID: 29461862 · PMCID: PMC6067107

23. **Cactus: Algorithms for genome multiple sequence alignment**
B. Paten, D. Earl, N. Nguyen, M. Diekhans, D. Zerbino, D. Haussler
Genome Research (2011-06-10) https://doi.org/bk4697
DOI: 10.1101/gr.123356.111 · PMID: 21665927 · PMCID: PMC3166836

24. **Contrasting evolutionary genome dynamics between domesticated and wild yeasts**
Jia-Xing Yue, Jing Li, Louise Aigrain, Johan Hallin, Karl Persson, Karen Oliver, Anders Bergström, Paul Coupland, Jonas Warringer, Marco Cosentino Lagomarsino, … Gianni Liti
Nature Genetics (2017-04-17) https://doi.org/f93kpp
DOI: 10.1038/ng.3847 · PMID: 28416820 · PMCID: PMC5446901

25. **Assemblytics: a web analytics tool for the detection of variants from an assembly**
Maria Nattestad, Michael C. Schatz
Bioinformatics (2016-06-17) https://doi.org/f9c485
DOI: 10.1093/bioinformatics/btw369 · PMID: 27318204 · PMCID: PMC6191160

26. **Discovery, genotyping and characterization of structural variation and novel sequence at single nucleotide resolution from de novo genome assemblies on a population scale**
Siyang LiuShujia Huang, Junhua Rao, Weijian Ye, Anders Krogh, Jun Wang
GigaScience (2015-12) https://doi.org/f75r4n
DOI: 10.1186/s13742-015-0103-4 · PMID: 26705468 · PMCID: PMC4690232

27. **Minimap2: pairwise alignment for nucleotide sequences**
Heng Li
Bioinformatics (2018-05-10) https://doi.org/gdhbqt
DOI: 10.1093/bioinformatics/bty191 · PMID: 29750242 · PMCID: PMC6137996

28. **The Pancancer Analysis of Whole Genomes (PCAWG).**https://dcc.icgc.org/pcawg

29. **Genomics England 100,000 Genomes Project.**https://www.genomicsengland.co.uk

30. **Whole Genome Sequencing in the NHLBI Trans-Omics for Precision Medicine (TOPMed).**https://www.nhlbiwgs.org/

31. **Sequencing and de novo assembly of 150 genomes from Denmark as a population reference**
Lasse Maretty, Jacob Malte Jensen, Bent Petersen, Jonas Andreas Sibbesen, Siyang Liu, Palle Villesen, Laurits Skov, Kirstine Belling, Christian Theil Have, Jose M. G. Izarzugaza, … Mikkel Heide Schierup
Nature (2017-07-26) https://doi.org/gbpnnx
DOI: 10.1038/nature23264 · PMID: 28746312

32. **Toil enables reproducible, open source, big biomedical data analyses**
John Vivian, Arjun Arkal Rao, Frank Austin Nothaft, Christopher Ketchum, Joel Armstrong, Adam Novak, Jacob Pfeil, Jake Narkizian, Alden D Deran, Audrey Musselman-Brown, … Benedict Paten
Nature Biotechnology (2017-04) https://doi.org/gfxbhs
DOI: 10.1038/nbt.3772 · PMID: 28398314 · PMCID: PMC5546205

33. **Bcftools 1.9**https://samtools.github.io/bcftools/

34. **A framework for variation discovery and genotyping using next-generation DNA sequencing data**
Mark A DePristo, Eric Banks, Ryan Poplin, Kiran V Garimella, Jared R Maguire, Christopher Hartl, Anthony A Philippakis, Guillermo del Angel, Manuel A Rivas, Matt Hanna, … Mark J Daly
Nature Genetics (2011-04-10) https://doi.org/d9k453
DOI: 10.1038/ng.806 · PMID: 21478889 · PMCID: PMC3083463

35. **Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications**
Andy RimmerHang Phan, Iain Mathieson, Zamin Iqbal, Stephen RF Twigg, Andrew OM Wilkie, Gil McVean, Gerton Lunter
Nature Genetics (2014-07-13) https://doi.org/f6b6dk
DOI: 10.1038/ng.3036 · PMID: 25017105 · PMCID: PMC4753679

36. **BEDTools: a flexible suite of utilities for comparing genomic features**
Aaron R. Quinlan, Ira M. Hall
Bioinformatics (2010-01-28) https://doi.org/cmrms3
DOI: 10.1093/bioinformatics/btq033 · PMID: 20110278 · PMCID: PMC2832824

37. **Mash: fast genome and metagenome distance estimation using MinHash**
Brian D. Ondov, Todd J. Treangen, Páll Melsted, Adam B. Mallonee, Nicholas H. Bergman, Sergey Koren, Adam M. Phillippy

Genome Biology (2016-06-20) https://doi.org/gfx74q
DOI: 10.1186/s13059-016-0997-x · PMID: 27323842 · PMCID: PMC4915045

38. **PHYLIP - Phylogeny Inference Package (Version 3.2).**
Joel Felsenstein
Cladistics (1989)