Comparing the performance of variant calling algorithms based on HiFi reads

Master's Thesis

Matúš Mikuš

Comparing the performance of variant calling algorithms based on HiFi reads

THESIS

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science

by

Matúš Mikuš

Student ID: 5144914

May 30, 2022

Pattern Recognition and Bioinformatics
Department of Intelligent Systems
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl

Preface

I would like to thank my daily supervisor Dr. Ahmed Mahfouz for his guidance, support and patience in what turned out to be a very challenging period, marked by the Coronavirus pandemic.
Furthermore, I would like to extend my thanks to Dr. Marcel Reinders for his oversight and useful feedback at crucial points in the thesis, as well as his agreement to join the thesis committee.
In a similar vein, I would like to thank Dr. Zaid Al-Ars, for his agreement to join the committee as an external member.
Next, I want to thank all the members of the Bioinformatics community with whom I have exchanged email conversations, and who provided useful feedback and guidance, as well as the counselling team at TU Delft for helping me navigate the challenges of work from home.
Finally, I would like to thank my family and friends for their online and offline support during my study at TU Delft.

Matúš Mikuš
Rijswijk, the Netherlands
May 30, 2022

Thesis Committee

Professor Dr. Marcel Reinders, Chair
Professor Dr. Ahmed Mahfouz, Daily Supervisor
Professor Dr. Zaid Al-Ars

Table of Contents

Abstract

Variant calling is a challenging, multi-stage process that allows researchers to compare genomes and their genetic variation at both the individual nucleotide level and the kilobase scale level. Variant calling is done by collecting genome data in the form of reads, processing the reads, phasing, or genotyping (for diploid genomes) and finally comparing the reads to a reference genome. The final product of variant calling is a list of differences, or variants, between the reference and each sequenced genome. With the introduction of long-read sequencing, in particular Pacific Biosciences HiFi reads, human genome variant calling can now be done efficiently and accurately, even on genomic regions that exhibit large amounts of variation and repetition, which is a difficult task for most variant callers. Two such genomic regions are the Major Histocompatibility Complex and Immunoglobulin Heavy Chain, which are both associated with the immune system. The aim of this study was to construct a pipeline that compares three publicly available, state-of-the-art variant calling algorithms for HiFi reads, DeepVariant, PBSV and Sniffles. The constructed pipeline was used to collect reads from twelve publicly available genomes, align them to a reference genome and call variants using all three variant calling algorithms. We have compared the variants on a global scale, across individual chromosomes, in specific difficult regions, compared the variation length, region of origin and type of variation. Furthermore, we have compared the individual algorithms' specificity and sensitivity, using already existing benchmarks. The results point towards the dominant performance of DeepVariant, however, for large variation, this algorithm is outperformed by PBSV. Unsurprisingly, all three algorithms struggled in repetitive and low-coverage regions, but often in different places, pointing towards the benefit of combining the results from multiple variant callers into a final callset.

Introduction

The introduction of third generation genome sequencing techniques, in particular PacBio HiFi reads, created an upheaval in the world of genome sequencing as it expanded the space of sequencing possibilities. The new technology provides reads which are orders of magnitude longer than those obtained with Next Generation Sequencing (NGS) methods, and their error rates are competitively low, usually in fractions of a percent [1]. The increased lengths of reads also allow researchers to accurately sequence difficult and repetitive regions of the human genome. As the PacBio HiFi technology matures, more tools taking advantage of the technology are developed and the need for benchmarks and studies comparing those tools grows, as does the need for more stringent standards. One family of such tools is the variant callers, algorithms that process the reads of one or several genomes and output a list of sites where each input genome differs from a reference. Variant callers allow researchers to compare genomes, identify differences associated with individuals or populations and determine connections between disease and variation. A large number of competing variant callers have been developed or fitted towards HiFi reads, in order to take advantage of the longer reads [2]. These modern variant callers are based on varying methods, ranging from graph-based, to statistical and machine learning-based, and as a result, their performance and accuracy varies considerably, with run-times and memory requirements varying up to an order of magnitude. With a tool ecosystem that is young and not yet standardized, it can be difficult to assess the strengths and weaknesses of individual tools and decide on which ones to use for a project. To help address this problem of tool uncertainty, this study was designed with the aim to compare three different variant calling algorithms in their performance in calling variants in difficult genomic regions, speed of runtime, ease of use, and compatibility with other tools in the HiFi read analysis ecosystem. A second part of the study aimed to give a recommendation about which variant caller to use, depending on the use case.

The specific difficult regions of the human genome analyzed in the study are the Major Histocompatibility Complex (MHC) and the Immunoglobulin Heavy Chain (IGH). The MHC is a region in the human genome, coding for over 200 genes. These genes are collectively called Human Leukocyte Antigen (HLA) genes and they code for cell surface proteins involved in antigen processing, preventing the immune system from targeting its own cells [3]. The HLA genes also mediate communication between leukocytes and body cells, which is where their name comes from. The MHC is highly polymorphic, containing multiple variants of each gene in the population. In addition, due to its involvement in the immune system function, the region contains a large amount of structural variation across individuals [3]. The IGH region codes for the large subunit of antibodies, or immunoglobulins, proteins that seek out and neutralize pathogens by binding to their antigens. Since each antibody binds to a unique antigen, in order to recognize a variety of threats, there exists a large variety of antigen binding sites, a consequence of variation in the region [4]. The IGH region features a large amount of repetition and as such, is susceptible to a large amount of insertions and deletions [4].

Its genes are also polymorphic, with a large amount of allelic variation [5]. The diversity and repetition of both regions contributes to their difficulty, and their involvement in the immune system makes them worthy of individual investigation.

There have been several papers benchmarking the performance of different variant callers, especially in the more mature field of short-read data variant calling. The short-reads approaches are numerous and each specializes in detection of specific variants, given that no method is able to reliably identify all types and sizes of variation [6]. A partial solution to this problem is to use multiple methods and combine their outputs using an algorithm for determining the validity of a variant, but those methods fall short of reaching perfect results, and the most complex variation, such as long insertions or complex repetitions might not be discovered using short-read data [6]. With the maturation of long-read sequencing technology, the long-read data variant calling field is also due to mature, and new benchmarking challenges such as the PrecisionFDA Truth V2 have been instigated in order to compare the methodology and performance of long-read variant callers [7]. This study relates to the benchmarking challenge, by comparing the performance of three distinct variant callers and attempting to construct a meta variant caller by combining their results.

The major questions addressed in the study are: how do the variant callers differ globally on genomes, across criteria such as variant length, read-depth requirements and regional differences? How do these variant callers perform in specific difficult regions of the human genome? Given the algorithms' differences, what is a suitable strategy to employ these algorithms based on a project's specification, in order to best exploit the algorithm's comparative strengths and minimize their weaknesses? These research questions were addressed by constructing a data pipeline that processes HiFi reads of human genomes, aligns each one to a reference and uses three variant calling algorithms to create lists of variant calls per genome per algorithm. These results are collated, compared and analyzed using a suite of variant call processing tools. A final analysis on the whole cohort of genomes attempts to give insight into the collected variation and provide direction for future work. A discussion on the results and research questions closes out the study.

Methods

Overview

We collect 12 publicly available genomes sequenced with the PacBio technology and available in the form of HiFi reads. The reads are pre-processed and sorted according to specific algorithm requirements, aligned to a reference using pbmm2, and three distinct variant calling algorithms are used to call structural variants. The results are restricted to chromosomes 6 and 14, single nucleotide polymorphisms (SNPs) are filtered out and various variant analysis tools are applied on the result files for generating statistics on the results. Command line tools such as vcf-tools and bcf-tools are used to generate further statistics such as number of variants of different lengths, or for isolating variants in various regions of the genome. After individual analysis, the three callsets are merged into union and intersection sets and further analysis is done on the global level, such as comparisons with benchmarks and publicly available structural variation lists.

Data acquisition and preprocessing

We take PacBio HiFi reads from three different sources: the Human Pangenome Reference Consortium (HPRC) which contains 29 PacBio sequenced human genomes at the time of writing [8], the Human Genome Structural Variation Consortium (HGSVC) which contains additional 7 human genomes [9], and the Genome in a Bottle Consortium (GIAB), containing final 7 genomes [10]. In total, we have obtained 43 genomes sequenced with the PacBio HiFi technology. For the final analysis, in the interest of time, computational efficiency and procedure simplicity, a smaller subset of the genomes was used and the majority of analysis was done on 12 genomes. Some of these, the HG001-HG007 were chosen due to the existence of previous analysis on those genomes, allowing for benchmarking and validation of the results of this study, others were chosen from a second source without extensive consideration, simply selecting genomes that all three variant callers managed to process. The data sources and scripts used for downloading are included in the Appendix. The reads are collated and aligned to a publicly available GRCh38.p13 reference genome without alternative alleles [11]. We align, sort and index the reads using pbmm2, a PacBio-developed wrapper for the Minimap2 aligner, using the CCS (Circular Consensus Sequencing) preset. The aligned reads are merged and optionally sorted again using the samtools toolset and set as input for the variant calling algorithms. Each individual variant calling algorithm has a slightly different procedure described below.

Variant Calling Algorithms

DeepVariant

The DeepVariant algorithm is based on a Convolutional Neural Network (CNN) architecture originally designed to classify images [12]. DeepVariant selects candidate variants from the input reads and translates each candidate region into a multi-channel image. Each channel encodes a different category of information: Read base, read quality, mapping quality, alignment strand, variant support from given reads and base difference from reference. Since DeepVariant is a deep learning model, it needs to be trained on data that most closely resembles the use case, and DeepVariant features a trained HiFi reads model. In order to increase accuracy, DeepVariant also supports the information about the read's haplotype, which is the information specifying which parent that read strand was inherited from. The haplotype information is provided using the WhatsHap phasing tool, assuming a diploid organism. In the precisionFDA Truth Challenge v2, DeepVariant scored among the best variant callers for difficult regions, with superior performance on insertions and deletions [13].

DeepVariant is combined of three sub-programs, make_examples, call_variants and postprocess_variants. The first sub-program converts input reads and the reference genome into native TensorFlow examples that can be evaluated by the deep learning model. This is a single-threaded operation, but its inputs and outputs can be sharded, as the process can be parallelized across the genome. Call_variants processes the native examples and evaluates them using the learned deep learning model. This step can be parallelized, but performance does not scale linearly with added cores. Given the machine learning nature of this task, call_variants performs best on several CPU units combined with a strong GPU unit, and this step is the only one that benefits from a GPU unit. Finally, post_process variants combines outputs from the call_examples step, sorts the records and outputs the final VCF file. Since this step requires sorting of all available data, it requires single threading and a large amount of memory.

We have run the 1.1.0 version of DeepVariant, the latest version available at the start of the project, using the PACBIO trained model. Extra arguments for the *make_examples* step of the process were vsc_min_fraction_indels=0.1, which is a slightly lower insertion and deletion (indel) acceptance treshhold than the default 0.12. The DeepVariant pipeline consists of running DeepVariant once, to identify heterozygous sites in the genome. This is followed by a WhatsHap command to phase and haplotag the heterozygous sites, followed by another run of DeepVariant, with an extra haplotype channel. According to the authors of DeepVariant, this procedure improves total performance scores, mostly by reducing indel errors by 40% [12].

PBSV

PacBio structural variant calling and analysis tools (PBSV), is a set of tools designed specifically for PacBio reads. The algorithm detects variants by collecting and partially ordering the flanking regions of the candidate variants. These flanks are subsequently realigned using a breakpoint-aware aligner and PBSV calls a variant if the realigned event size and location are supported by the realignment. This procedure is similar for insertions, deletions and inversions. Translocations and duplications are detected by their specific signatures in the split reads [14]. PBSV features a range of flags that can be set in order to filter the variant calls to a desirable standard, as well as a HiFi read preset. To improve tandem repeat discovery, a feature common in centromere regions, the PBSV algorithm was supplemented by a tandem repeat annotation BED file, but this result points to a weakness of the algorithm in dealing with large-scale tandem repeats.

A major advantage of the tool is its simplicity, as it is designed to be usable by researchers without a large amount of experience with variant calling. The configuration allows for setting a large amount of flags, such as maximum lengths for various types of variation or result calling criteria based on reads and samples. A single PBSV run consists of a discover step, which filters out reads without structural variation signatures, followed by a variant calling step. For this study, the following parameters were used: --ccs -A 2 -O 2 -P 20 -m 10. The PBSV algorithm can be parallelized by processing each chromosome separately.

Sniffles

The last structural variant caller, Sniffles, was designed to optimize for performance given low available coverage. The Sniffles algorithm begins by estimating the parameters of the given set of reads, such as the values and distribution of alignment scores. This step is done in order to gain a global understanding of the dataset and informs further variant discovery steps. The algorithm proceeds by discarding unreliable reads, such as reads with low mapping quality, with a low ratio of best and second-best alignment scores (indicating uncertainty about the true alignment), or simply a read with the best alignment score under a certain threshold. After the read filtering step, Sniffles scans the alignments, looking for noisy regions or alignments that show an increased indel or mismatch rate, indicating the presence of a structural variant. The actual variant calling step happens in multiple steps and candidates are generated according to various parameters, such as length, coverage quality, variant type and position in the genome. Potential variants are clustered, filtered, with extra steps for nested variants. Due to its design and the presence of a complex ruleset, Sniffles is able to detect nested structural variation, a feature useful in regions difficult to parse. The algorithm's authors also tested Sniffles' performance on downsampled genomes, obtaining satisfactory performance even at 15x coverage, with approximately 80% precision and 70% recall on variants of all types [15]. Sniffles can further be parallelized by running each chromosome separately, with a varying number of threads per chromosome.

Sniffles requires sorted aligned reads, and this step is accomplished using samtools. The parameters used for Sniffles in this project were: --min_support 2 -l 10 --genotype –skip_parameter_estimation. The low read support requirement was subsequently addressed in the variant call post-processing step.

Variant merging

As described above, each variant calling algorithm performs better for certain classes of variation than other classes, and these strengths are often complementary, which gives a strong reason for combining their results in a meta-caller. This task was accomplished using the SURVIVOR tool. For each genome, two callsets are created, a union callset containing each variant from each pipeline, filtered for overlaps within a range of 1000bp. This is a lenient callset, sacrificing precision for recall, as it contains false positives from all three algorithms. Another constructed alternative is a stringent intersection callset, which requires the support of all three variant callers in order to include a variant. This callset is likely to have higher precision, at the cost of recall, as it might miss a correct variant called by a single caller, or a pair of callers.

Validation and Evaluation

In order to validate the algorithms present in this study, we utilize well-studied genomes and compare the results from this study to a previous study done on the same genomes, comparing the final variant file results. A metric used for validation is total overlap between the three variant callers and the benchmark, as well as individual overlaps. To this effect, we have obtained a publicly available variant callset for the genome HG003 from the Genome in a Bottle (GIAB) online repository [16]. This callset was generated with PacBio HiFi reads and the variants were called using the GATK and DeepVariant callers, and thus substantive overlap between the official callset and the generated DeepVariant callset is unavoidable.

An additional evaluation step followed the creation of the total variant callset. We have used the dbVar Human Nonredundant Structural Variants database of verified insertions and deletions [17]. There was a misalignment of the way the variant lists report insertions with the database’s reporting, the two files agreed on the start position but not the end position, and analysis tools reported minimal overlaps, even with generous buffer parameters. Thus, only the results for deletions are summarized in the results section.

Finally, the obtained variant calls from all three callers were manually inspected in the Integrated Genomics Viewer. In total, approximately 200 variant calls were observed and the differences in results were manually analyzed and documented in the Appendix. This analysis was focused on differences between variant callers and algorithm-informed explanations for these differences are provided.

Results

Overview

We used three variant callers (DeepVariant, Sniffles, PBSV) to detect variants from HiFi sequencing data from 12 genomes. These genomes had an average sequencing coverage of 30-32x for the HG001-HG007 genomes and at least 35x for the remaining genomes. In total we detected 56653 non-redundant variants from 12 genomes, the full breakdown is described in Table 1. The variant calls were further split into various categories and analyzed using vcf processing tools and manual inspection. What follows is a selection of research questions that compare the performance of the algorithms in question on various criteria.

Table 1: The breakdown of total variants per genome and the union of all found variants and variants found by at least two algorithms. Of note is the unusual performance on HG006, where Sniffles found significantly more variants than usual, with lower overlap to the results of the other algorithms.

Genome	DeepVariant	PBSV	Sniffles	Union	Intersection
HG001	13046	9737	8550	15544	6990
HG002	13831	10043	8984	16186	7498
HG003	13808	10099	8379	16134	7212
HG004	13884	10157	8089	16076	7001
HG006	13806	10122	14129	24005	3661
HG007	13799	10322	8147	16142	7012
HG00438	13402	10173	9252	16046	7605
HG00673	13531	10142	9244	16146	7690
HG00732	13548	10089	9419	16277	7672
HG00735	13890	10448	9564	16559	7886
HG00741	13968	10544	16440	22423	8195
HG01071	13131	10107	8961	15748	7450

From all variants called by the three algorithms, using the ‘bcftools isec’ command on the vcf file intersections of individual genomes, we have found that 8689 variants appeared in at least three samples and 3059 variants appeared in at least eight samples.

The performance of the three algorithms differed greatly. The DeepVariant version used was 1.1 and the pipeline took the most time from the three implemented pipelines. Limiting the algorithm to chromosomes 6 and 14 requires approximately 12 hours on 8 TU Delft cluster CPU cores. This is a consequence of the requirement to run the main algorithm twice with a phasing step between those runs, as well as not using a GPU unit, which would half the time requirements of the algorithm [18]. The PBSV version used was 2.6 and its total runtime for a single genome on 8 TU Delft cluster CPU

cores was approximately 2 hours. The Sniffles version in this study was 1.0.12a and the runtime for a single genome on 8 TU Delft cluster CPU cores was approximately 2 hours.

General performance

The different variant calling algorithms have slightly different ways of annotating variants, which complicates the process of variant calling. Generally, the same variants called by sniffles had their position index lower by one and thus detailed comparisons and metrics generation of unique variants had to include offsets for each variant. Manual inspection of the reads and variants using the Integrated Genome Viewer (IGV), as well as a SURVIVOR overlap function confirmed that the majority of reads were called non-uniquely by all three algorithms.

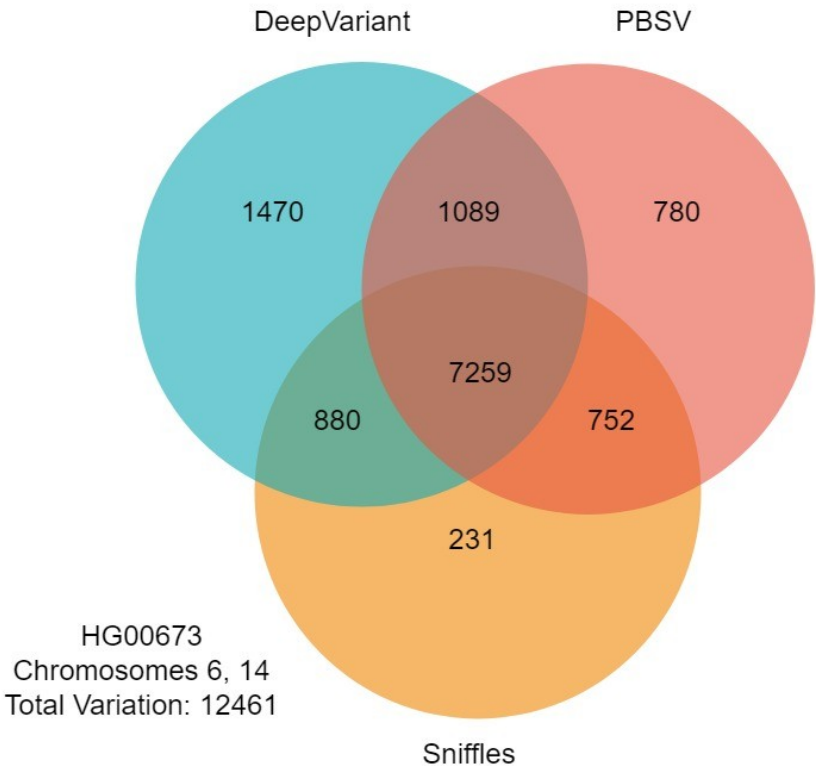


Figure 1: Venn diagram for variant overlaps between different algorithms

Using the vcf-compare tool from vcftools for the filtered reads for the HG00673 genome, the result displayed in Figure 1 is that 67.9% of Deepvariant variants, 73.5% of PBSV variants and 79.6% of Sniffles variants are shared for all three variant callers and only 13.7%, 7.9% and 2.5% of variants respectively are unique. With regards to the unique variants, Deepvariant manages to find the most.

This can be seen specifically in areas with duplication and repetition, but there are also genomic instances with a large amount of repetition and duplication which DeepVariant does not call as variation, see the Appendix for examples. A frequent occurrence is when an insertion/deletion is supported by only one strand. In these situations both PBSV and DeepVariant disagree on whether an event is a variant or not, such as in Figure 2. Sniffles in general calls these events as variants.

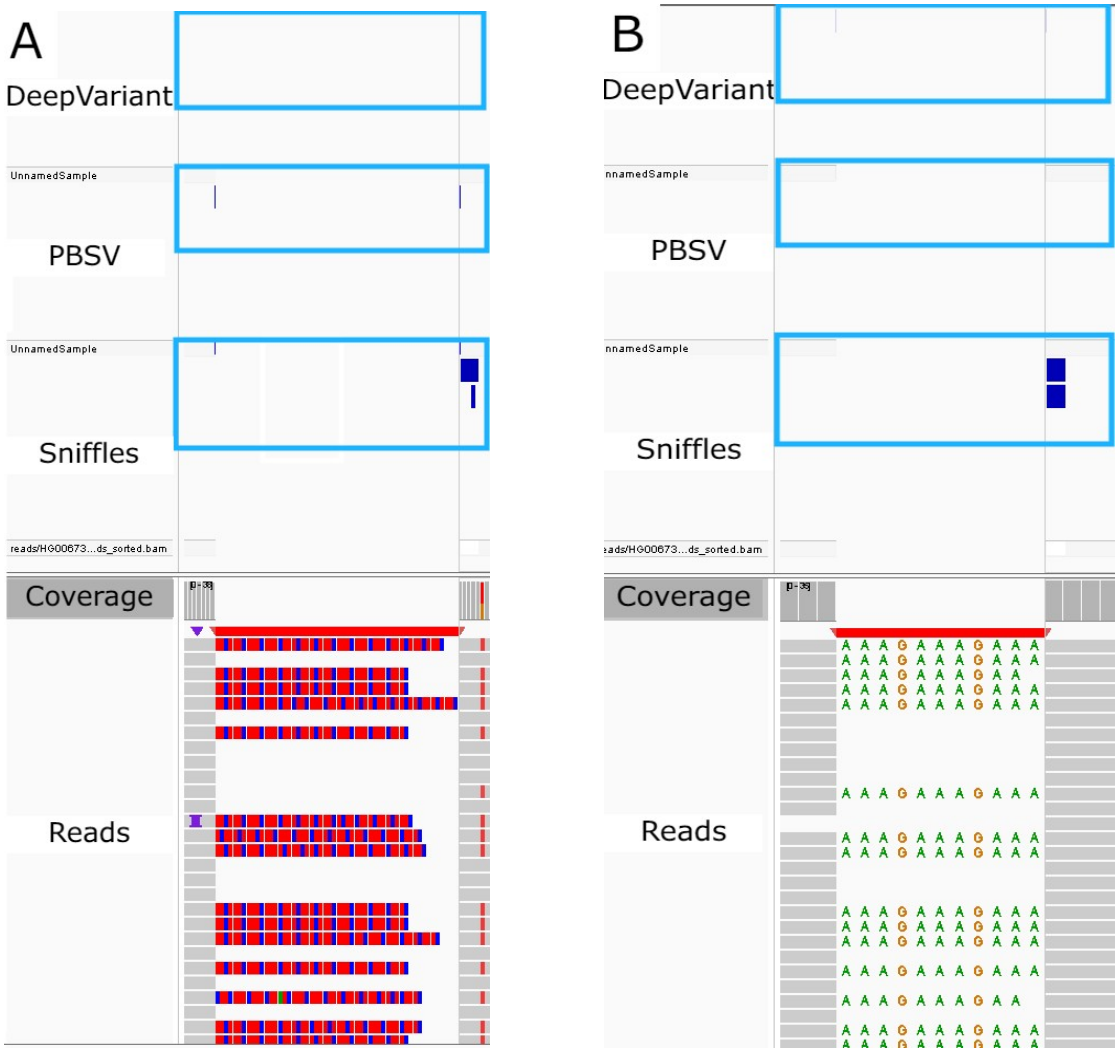


Figure 2: DeepVariant (A) and PBSV (B) respectively missing a repetitive region. In the Reads section of each image, each nucleotide is colored with a separate color. Each blue box shows the events called by the respective algorithm. Figure 2A shows a highly repetitive strand which is called as an insertion by Sniffles and PBSV, while DeepVariant ignores the event. Figure 2B shows a similar occurrence, where the short region is composed of three repetitions of the AAA chain, followed by a G nucleotide. This event is ignored by PBSV.

Results

Sniffles and PBSV often agree on large insertions, despite their different annotation. A flaw of Sniffles is its calling of extremely large events as shown in Figure 3: an 800kbp event called as an insertion, which is a false positive, despite support from at least three reads. We can postulate this to be caused by the low read support requirement as well as the lower basepair threshold for calling a variant, as Sniffles considered significantly more read events than the algorithm does by default.

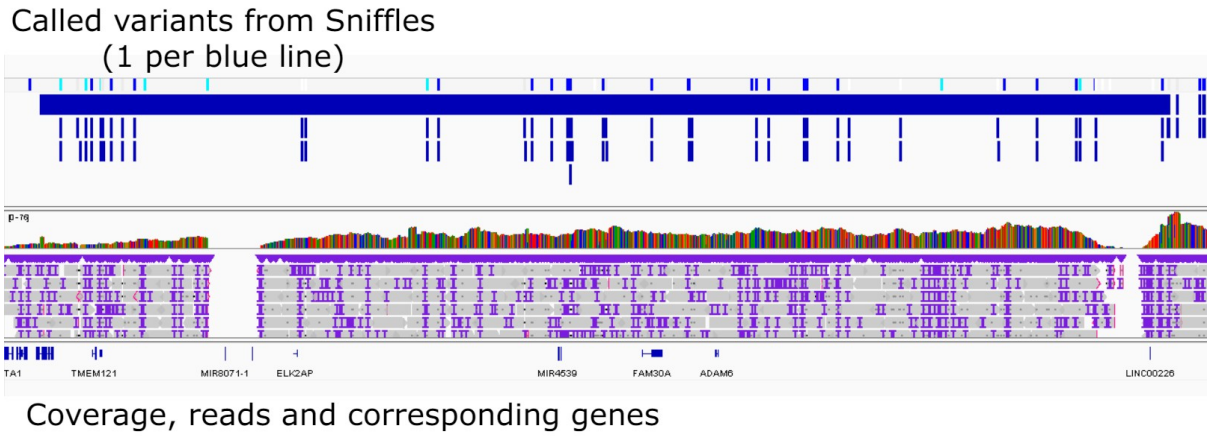


Figure 3: Sniffles calling an 800kbp ‘insertion’. The long bar at the top of the screenshot is an indicator of a single event, however, this event is unlikely to be an insertion due to the read support profile in the middle of the screenshot. Only Sniffles called events such as this one as variants.

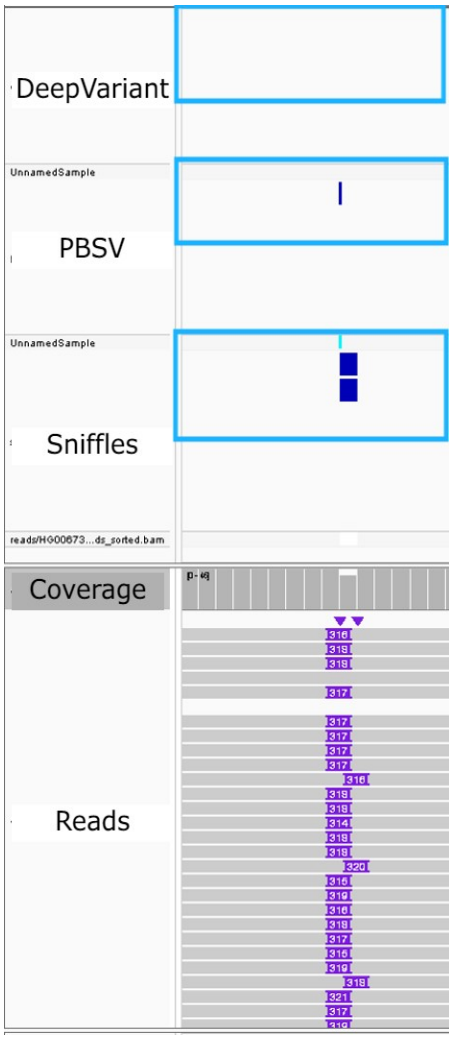


Figure 4: DeepVariant not calling a 317bp long insertion, despite all reads and the other algorithms supporting that conclusion. Each blue box shows the events called by the respective algorithm.

DeepVariant called smaller variants, compared to Sniffles and PBSV. This can be due to a lower amount of training data on large variants, as well as the slightly more conservative setting of the algorithm. On large variants, Sniffles and PBSV algorithms tended to agree, which points to lower sensitivity of DeepVariant.

Comparing variant calls to a ground truth

In order to validate the performance and results of the procedure in the study, we have obtained a publicly available variant callset for the genome HG003 from the Genome in a Bottle (GIAB) online repository. This callset was generated with PacBio HiFi reads and the variants were called using the GATK and DeepVariant callers, and thus substantive overlap between the official callset and the

generated DeepVariant callset is unavoidable. Nevertheless, the resulting overlaps between the obtained callset and the other two callsets generated from our PBSV and Sniffles were satisfactory. From 3894 variants longer than 10bp on chromosomes 6 and 14, 3190 (81.9%) were also called by at least one of the three examined variant callers, with 2117 (54.4%) called by all three. Only 96 variants found in the benchmark were not replicated by the pipeline, with the remaining 608 variants filtered out due to grouping. Excluding the grouped variants, 97% of variant calls were replicated by this study's algorithms and thus the validation step was successful.

Characterizing the length of variant calls

We have investigated whether the different algorithms differ in the lengths of their called variants. To this end, we have collected the total list of variants for each individual algorithm. From these three lists, the read lengths of individual variants were extracted and plotted in a histogram. The majority of structural variants reported by DeepVariant are up to 3,000 bp in length. PBSV reports the majority of its variants up to 4,000 bp. Sniffles has the capacity to report large variants up to 30,000 bp, but the higher propensity for reporting large variants also causes the algorithm to report deletions up to 6mbp that turn out to be false positives upon manual inspection. After filtering variants with support from less than 5 reads from Sniffles, the resulting variants called appear in the 5,000 – 7,000 bp length range.

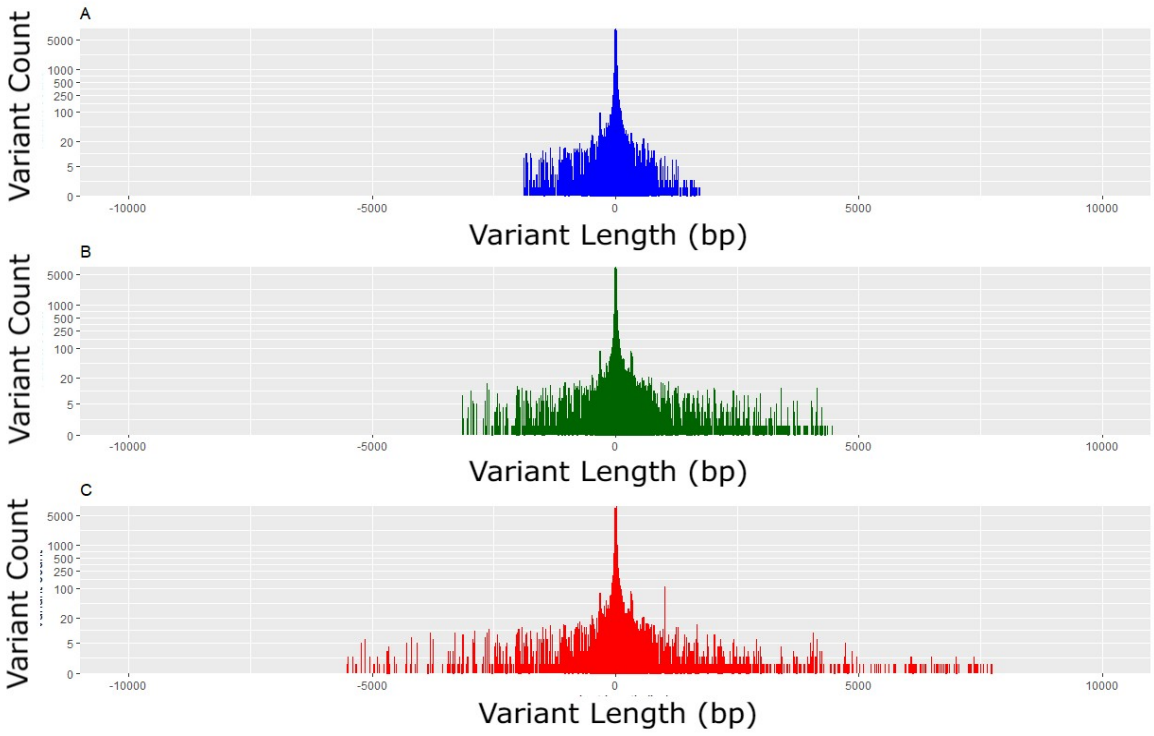


Figure 5: Total length distributions for all variants called by the three algorithms in question, after filtering for outliers. DeepVariant (A), PBSV (B), Sniffles (C).

Performance across different genomic regions

Next, we investigated if the variants calls differed in different types of genomic regions. We have obtained a demarcation of genomic region types for the reference genome, GRCh38.p13, and used it to create variant files with variants from singular region types. In these files, we have compared the number of variants across repeat regions, gaps, centromeres, coding, and non-coding regions. All three variant callers reported similar variation for repeat, coding and non-coding regions. A small difference can be seen in assembly gap regions, where DeepVariant did not find variants, while other variant callers did. This could be seen as a feature of the DeepVariant algorithm, which recognizes these regions as low confidence and excludes them from the analysis, or a feature of the remaining variant calling algorithms where the decision is to include variants, even though they might be of lower confidence. On the other hand, DeepVariant had a slightly higher rate of reporting variants in the centromere regions, but the difference is negligible.

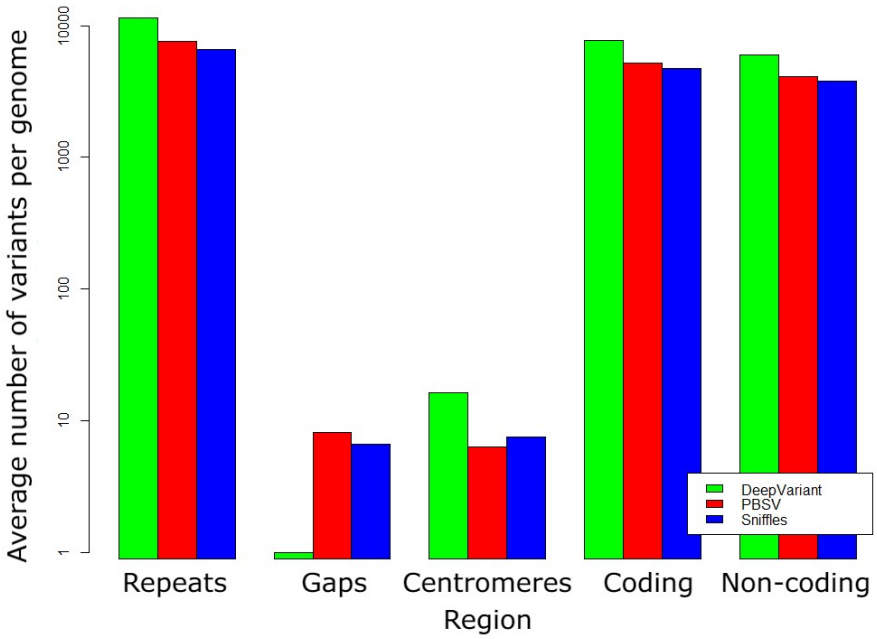


Figure 6: Differences in average variant numbers in different genomic regions.

Assessing the effect of read depth on variant calling

A relevant analysis point was to compare read depths of variants and their relationship to the number of variants called. Given Sniffles lack of reporting read depth in the resulting variant files, comparisons were drawn between DeepVariant and PBSV. Generally, DeepVariant reports higher read depths for its

variants. This is due to a filtering step in the PBSV procedure which filters away reads that are not relevant in calling structural variants, as the underlying alignment of both algorithms is to the same reference genome. Given the slightly lower number of variants called by PBSV, this can be seen as a weakness of the algorithm, which sacrifices sensitivity for specificity.

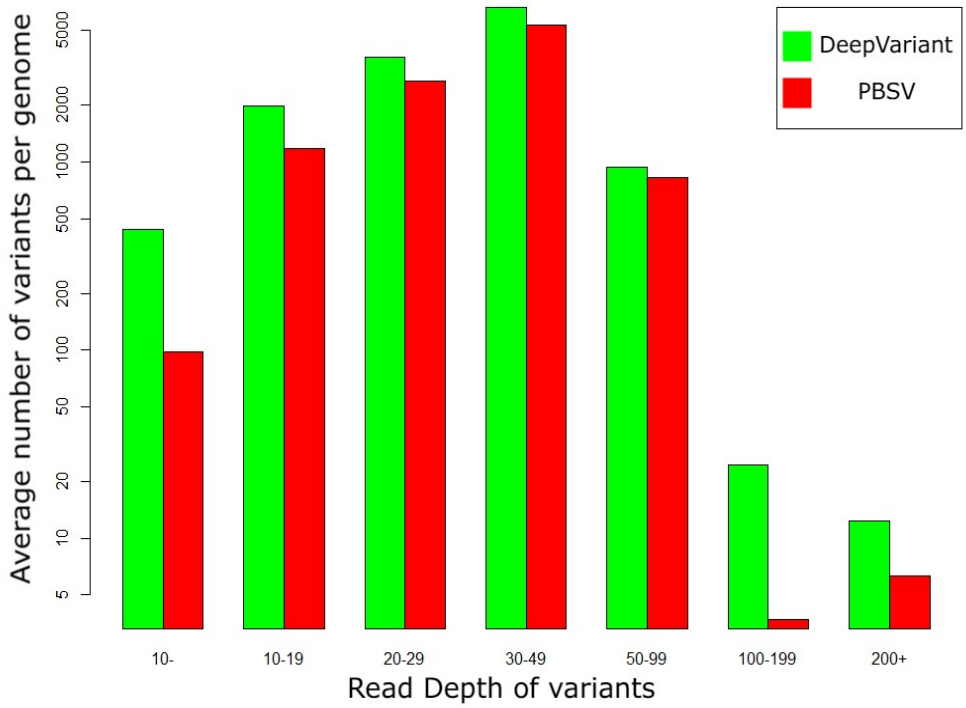


Figure 7: Read Depth differences between DeepVariant and PBSV. PBSV calls less variants in general, but the largest difference is in variants with high read depths, since PBSV filters out reads that are irrelevant, lowering the average read depth of a variant.

Performance in calling indels

Indels are a relatively common and simple form of variation and this makes them easy to detect and mark. Despite their simplicity, they are highly important due to their involvement in disease [19]. In this study, since variation under 10bp is removed, indels are the most common form of variation, and this allows us to compare the final results without statistical interference inherent in low sample size events. As mentioned in a previous section, the dbVar database of structural variation was used to compare the algorithms' performance on deletions. The database contained 179,566 results after filtering and the total amount of captured variation in deletions are summarized in the following table. The results are created from the reported deletions of 12 genomes analyzed in this study.

Table 2: reciprocal overlaps between the benchmark BED file and the individual algorithm BED files, set at 90% and 99% respectively.

Caller	Called (90%)	Percentage (90%)	Called (99%)	Percentage (99%)
DeepVariant	21081	11.73%	7143	3.98%
PBSV	25290	14.08%	8383	4.67%
Sniffles	26974	15.02%	7381	4.11%
Total	28685	15.98%	8524	4.75%

The results in Table 2 show that at the lower reciprocal overlap criterion, the variation called by Sniffles is the largest of the three callers. The Sniffles result can be seen as a result of lower read support requirements for variants called by Sniffles, set at 5 reads required to consider a variant, as well as a presence of larger deletion calls. In comparison, the read requirements for PBSV were stricter (at least 20% of reads supporting each variant call). Finally, DeepVariant relies in part on training data, and it is reasonable to expect not all genomes from this comprehensive study are present in the training data, thus decreasing the number of variants called. We can conclude that the variants called by PBSV are better at finding the right number of deletions than the conservative DeepVariant algorithm and the exaggerated results of Sniffles, and this conclusion is supported by a larger number of variants considering the 99% overlap.

Variant calling in the IGH and MHC loci

As explained in the introduction, the regions coding for the IGH and MHC loci were chosen due to their difficulty, in part because of their repetitive nature. Repetition is a feature variant callers struggle with, and this experiment has been chosen to explore the responses of individual callers to such difficulty. To this end, 12 genomes were selected and the total variation in the IGH and MHC regions was collected. The genomic regions were obtained from the online National Center for Biotechnology Information database, mapping regions to equivalent genomic coordinates for the GRCh38.p13 reference. In particular, the regions chr6: 28,510,120 – 33,480,577, comprising 412 genes and containing the major HLA genes [20], and chr14: 105,586,437 – 106,879,844, comprising 205 genes and containing the immunoglobulin heavy chain locus, are selected [21]. The results collected here mirror those of the main section, notably the propensity for PBSV and Sniffles to call longer insertions, but new trends emerged too, such as lower concordance between the variant callers. This result was expected, given the difficult and repetitive nature of the regions. In total, 339 variants were called in total by the three algorithms, averaging to 30 variants per genome. Using vcf-compare as a benchmarking tool, we find that 114 variant calls are shared between all three callers, which comprises 70%, 50% and 44% of all calls for DeepVariant, PBSV and Sniffles respectively. DeepVariant found the least amount of unique variants, however, its variants were also replicated the most, with only 10% of it variants not supported by another caller. Conversely, 27% of variants reported by Sniffles were unique, and those were most often long variants, given Sniffles' propensity for calling long variants,

described previously. Using SURVIVOR to merge individual insertions and deletions, we obtain 158 insertions and 130 deletions, due to a stricter parameter configuration.

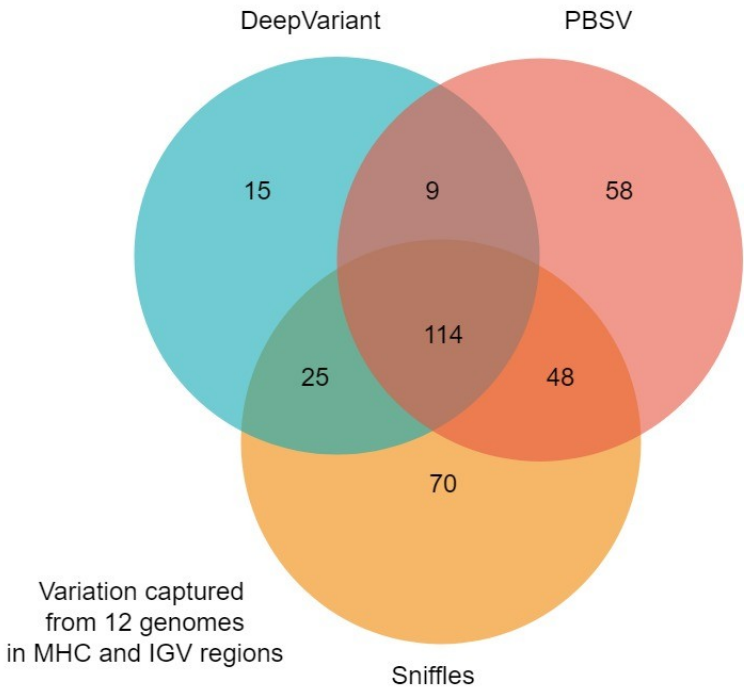


Figure 8: Total variation captured from 12 genomes in MHC and IGV regions.

When limiting the maximum length to 10,000bp, to avoid outliers, Sniffles and PBSV report higher rates of calling long variants in the MHC and IGH regions. Large indels are the main source of disagreement between the algorithms, and this remains in line with previous analysis on the global scale. What can be seen from the reads is that the coverage for these difficult regions is much lower, often dropping to 10x and all three algorithms struggle with calling accurate variants in such setting, even Sniffles, despite its design to handle such situations well. As a result, disparities in results are expected and ground truth difficult to ascertain. Interpolating from the whole genome results, we can assume that the Sniffles results are less sensitive, misreading low-coverage regions as large variation. On the other hand, DeepVariant exhibits a tendency to call fewer unique variants, a possible consequence of imperfect coverage of the training data.

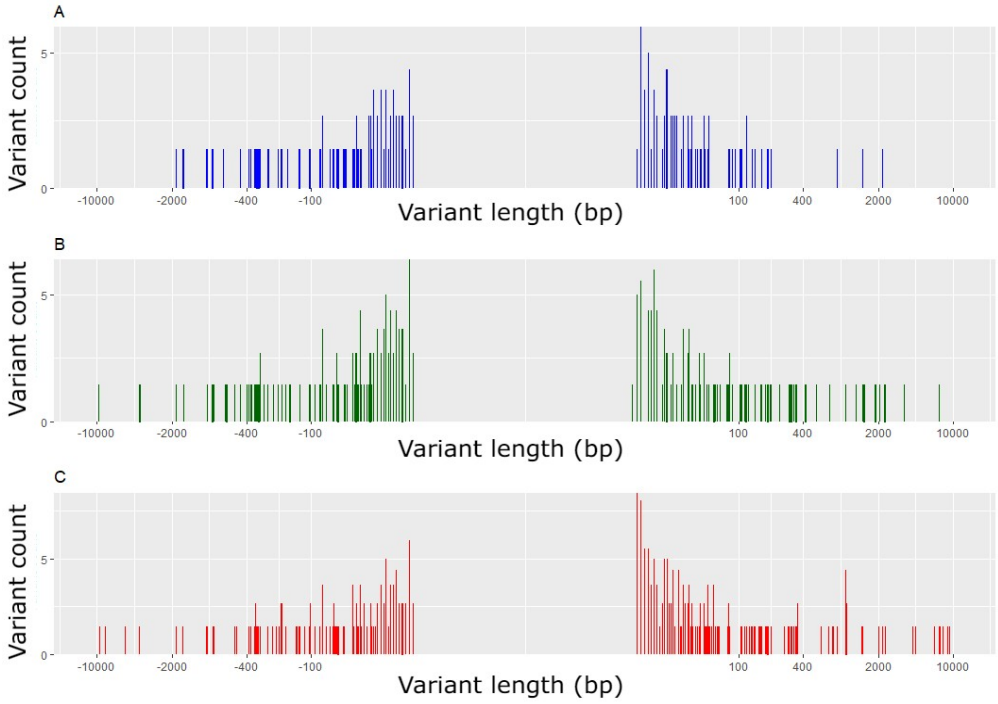


Figure 9: length distribution of MHC and IGH region variant calls. DeepVariant (A), PBSV (B), Sniffles (C). Of note is the length distribution that mirrors the global one, with Sniffles collecting most long variants, followed by PBSV then DeepVariant.

Discussion

In this study, we have used three different variant calling algorithms on 12 human genomes. We have validated the procedure of variant calling for each algorithm and compared their performance and difficulty of use. The resulting variant calls for each algorithm were compared across a suite of criteria, such as the performance in difficult regions, the ratio of insertions and deletions, the performance on long variants and the amount of total variation found.

DeepVariant is generally the best-performing variant caller

DeepVariant reports the largest percentage of unique variant calls, but only for relatively shorter base pair lengths. The remaining two variant callers do find longer variation more easily. Both of these factors can be influenced by parameter tuning of individual algorithms, but when considering a combination of uniqueness and accuracy, DeepVariant seems to perform the best. Sniffles called a lower amount of variation and its variants were less unique. This factor, combined with the different formatting of the outputs which complicated post-processing and analysis lowers the attractiveness of Sniffles. PBSV had the best validation results, overlapping with the largest amount of previously discovered variants. The final recommendation is to use DeepVariant as a primary variant caller, if the research objective is less time sensitive or if a GPU unit is available for the call_variants step. PBSV is a good algorithm for discovering longer variation, since DeepVariant called longer variants conservatively. PBSV is also a solid choice in situations where a fast, out-of-the-box solution is required. However, if the time requirement is an obstacle, or read coverage is not in the 20-30x range optimal for the remaining variant callers, Sniffles could serve as a compromise, if initialized with strict parameter settings to minimize the generation of large false positives.

Table 3: Comparison of the variant callers on different qualitative features

Criterion/Caller	DeepVariant	PBSV	Sniffles	Notes
Ease of use	Medium	Easy	Easy	
Tool compatibility	Good	Good	Poor	
Compute requirements	High (Medium with a GPU)	Low	Low	
Number of variants in difficult regions	Lowest	Medium	Highest	
Average amount of called variation	Highest	Medium	Lowest	
Length distribution	Short	Medium	Long	Sniffles outliers due to parameters
Indel validation	Good	Great	Good	

Algorithm output differences

Combining the output of multiple variant callers was at times a task that well illustrates the diverse state of sequencing technology tool landscape. Each variant caller was optimized to work best with a different alignment algorithm – DeepVariant with minimap2, PBSV with pbmm2 (minimap2 wrapper) and Sniffles with NGMLR. We have implemented pbmm2 as the default alignment software which could have had negative implications on the performance of Sniffles. The resulting VCF callsets also differed in the information provided. Sniffles was notoriously difficult to work with, not providing essential INFO columns in the VCF format specification. The caller also did not indicate the reference nucleotide for each variant, and lacked the read depth (DP) INFO column, which was a useful feature that would help distinguish the performance of Sniffles from the other two variant callers, especially in difficult-to-parse regions, which often feature lower read depth support.

The final output of Sniffles was also incompatible with vcftools, a suite of tools used for analysis of VCF callsets, further illustrating the variety of tools and standards used by different researchers in tool development. When it came to individual analysis, the difference in vcf output notation proved to be a complicated task when using out-of-the-box tools, such as bcftools isec for creating intersections. With a lack of support for offsets in the bcftools command and the generated offsets in Sniffles, the resulting intersections of VCF files were inaccurate and tools that respected take offsets into considerations had to be used. The differing algorithms often disagreed in the VCF file, even when they reported the same event. Individual inspection managed to catch some of these cases, and while they are not numerous, an intelligent filtering step would be beneficial in filtering out different descriptions of the same event.

Finally, all three algorithms were updated since the experiments in this project were conducted. Sniffles in particular had a major update which claims to have improved the performance of the variant caller, as well as expanding the amount of information given for the variants. This update could bring the performance of Sniffles closer to the remaining variant callers. Both PBSV and Sniffles had relatively similar performance in a 2019 variant caller performance study, with precision scores around 70% [22], but both tools have been updated since then, and their performance is currently expected to be better.

Limitations and direction for further work

Some decisions made in the study design turned out to limit the results and hinder analysis, and they warrant a short discussion. One such decision was not completely validating the variant calling protocol and analysis for individual algorithms, before running the experiments on multiple genomes. The parameter tuning for Sniffles, for example, was done quite early in the pipeline design process, and only comparisons between final results uncovered the need for re-tuning of the parameters, in order to reduce the number of false positives called by Sniffles. By that point, all the genomes were processed with Sniffles and re-processing was not feasible. In a similar vein, the large size of each genome, for both unaligned and aligned versions, combined with limited storage space prevented the possibility of re-aligning the genomes and repeating the variant calling, since the processed genomes were

periodically removed, to free up space for more genomes. Availability of more storage space would have solved this problem.

In terms of designed limitations, the study was also designed to exclude all SNPs and variation under 10bp, both due to tool limitations and to focus on larger-scale variants. Including SNP analysis and understanding correlations between SNPs and structural variants in individual genomes could illuminate the relationship between the variant types and disease, since the difficulty in calling low-coverage regions is further increased by interference from SNPs.

Finally, downstream analysis could be expanded to additional difficult regions in the human genome, such as centromeres or the killer cell immunoglobulin-like receptor region. The analysis of difficult genomic regions using sequencing data remains an open problem in genomics and insights into the patterns in repetitive regions would improve understanding of the relationships between structural variation and disease.

Conclusion

This project compared three different variant calling algorithms, benchmarked their performance and analyzed their results. The project also highlighted the benefits and drawbacks of calling variants from HiFi reads, primarily the gains in performance caused by using Machine Learning methods, the challenges of comparison caused by different protocol decisions made by algorithm developers and the strictness and incompatible requirements of some tools in the variant calling ecosystem. With the continual improvements in individual algorithms, their performance continues to converge, and the final decision on which individual tool, or combination of tools to use will depend partly on the size of the dataset, the available time and compute, and the nature of the problem. DeepVariant did have the best performance, thanks to its modern, Machine Learning based approach, but it also required the most compute and time per genome, due to not using a GPU unit. It also struggled with calling longer variants, where PBSV did better. PBSV was simpler to use, its results overlapped significantly with DeepVariant, albeit with slightly less unique results. Its compute and time requirements were also significantly lower than DeepVariant's. Finally, Sniffles was the fastest to run and had low coverage requirements. However, its output could be considered the worst in terms of false positives, or unique variants reported, with the exception for difficult regions, where analysis is complicated due to low coverage. Moreover, the file formatting standards for Sniffles differed from the remaining two algorithms which complicated downstream analysis. Overall, the algorithm to use depends on the context, but as a general recommendation, PBSV strikes a balance between fast performance and quality results, while DeepVariant is optimal to use if result quality is of utmost importance.

Bibliography

[1] T. Hon *et al.*, “Highly accurate long-read HiFi sequencing data for five complex genomes,” *Sci. Data 2020 71*, vol. 7, no. 1, pp. 1–11, Nov. 2020, doi: 10.1038/s41597-020-00743-4.

[2] M. Mahmoud, N. Gobet, D. I. Cruz-Dávalos, N. Mounier, C. Dessimoz, and F. J. Sedlazeck, “Structural variant calling: The long and the short of it,” *Genome Biology*, vol. 20, no. 1. BioMed Central Ltd., p. 246, Nov. 20, 2019, doi: 10.1186/s13059-019-1828-7.

[3] “The major histocompatibility complex and its functions - Immunobiology - NCBI Bookshelf.” https://www.ncbi.nlm.nih.gov/books/NBK27156/ (accessed May 03, 2022).

[4] C. T. Watson and F. Breden, “The immunoglobulin heavy chain locus: Genetic variation, missing data, and implications for human disease,” *Genes and Immunity*, vol. 13, no. 5. Nature Publishing Group, pp. 363–373, Jul. 03, 2012, doi: 10.1038/gene.2012.12.

[5] A. A. Bashirova *et al.*, “Population-specific diversity of the immunoglobulin constant heavy G chain (IGHG) genes,” *Genes Immun.*, vol. 22, no. 7–8, p. 327, Dec. 2021, doi: 10.1038/S41435-021-00156-2.

[6] M. Mahmoud, N. Gobet, D. I. Cruz-Dávalos, N. Mounier, C. Dessimoz, and F. J. Sedlazeck, “Structural variant calling: The long and the short of it,” *Genome Biol.*, vol. 20, no. 1, pp. 1–14, Nov. 2019, doi: 10.1186/S13059-019-1828-7/TABLES/2.

[7] N. D. Olson *et al.*, “PrecisionFDA Truth Challenge V2: Calling variants from short and long reads in difficult-to-map regions,” *Cell Genomics*, p. 100129, Apr. 2022, doi: 10.1016/J.XGEN.2022.100129.

[8] “human-pangenomics/HPP_Year1_Assemblies: Assemblies from HPP Year 1 production.” https://github.com/human-pangenomics/HPP_Year1_Assemblies (accessed May 03, 2022).

[9] “HGSVC2 | IGSR data collection.” https://www.internationalgenome.org/data-portal/data-collection/hgsvc2 (accessed May 03, 2022).

[10] “genome-in-a-bottle/giab_data_indexes: This repository contains data indexes from NIST’s Genome in a Bottle project.” https://github.com/genome-in-a-bottle/giab_data_indexes (accessed May 03, 2022).

[11] “GRCh38 - hg38 - Genome - Assembly - NCBI.” https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.26/ (accessed May 03, 2022).

Bibliography

[12] R. Poplin *et al.*, "A universal snp and small-indel variant caller using deep neural networks," *Nat. Biotechnol.*, vol. 36, no. 10, 2018, doi: 10.1038/nbt.4235.

[13] N. D. Olson *et al.*, "precisionFDA Truth Challenge V2: Calling variants from short- and long-reads in difficult-to-map regions," *Carlos Flores*, vol. 5, p. 21, Nov. 2020, doi: 10.1101/2020.11.13.380741.

[14] "PacificBiosciences/pbsv: pbsv - PacBio structural variant (SV) calling and analysis tools." https://github.com/PacificBiosciences/pbsv (accessed May 03, 2022).

[15] F. J. Sedlazeck *et al.*, "Accurate detection of complex structural variations using single-molecule sequencing," *Nat. Methods*, vol. 15, no. 6, pp. 461–468, Jun. 2018, doi: 10.1038/s41592-018-0001-7.

[16] "genome-in-a-bottle/giab_data_indexes: This repository contains data indexes from NIST's Genome in a Bottle project." https://github.com/genome-in-a-bottle/giab_data_indexes (accessed May 04, 2022).

[17] "dbvar/Structural_Variant_Sets/Nonredundant_Structural_Variants at master · ncbi/dbvar." https://github.com/ncbi/dbvar/tree/master/Structural_Variant_Sets/Nonredundant_Structural_Variants (accessed May 03, 2022).

[18] P. J. Huang *et al.*, "DeepVariant-on-Spark: Small-Scale Genome Analysis Using a Cloud-Based Computing Framework," *Comput. Math. Methods Med.*, vol. 2020, 2020, doi: 10.1155/2020/7231205.

[19] M. Lin, S. Whitmire, J. Chen, A. Farrel, X. Shi, and J. T. Guo, "Effects of short indels on protein structure and function in human genomes," *Sci. Reports 2017 71*, vol. 7, no. 1, pp. 1–9, Aug. 2017, doi: 10.1038/s41598-017-09287-x.

[20] "HLA-A major histocompatibility complex, class I, A [Homo sapiens (human)] - Gene - NCBI." https://www.ncbi.nlm.nih.gov/gene/3105 (accessed May 03, 2022).

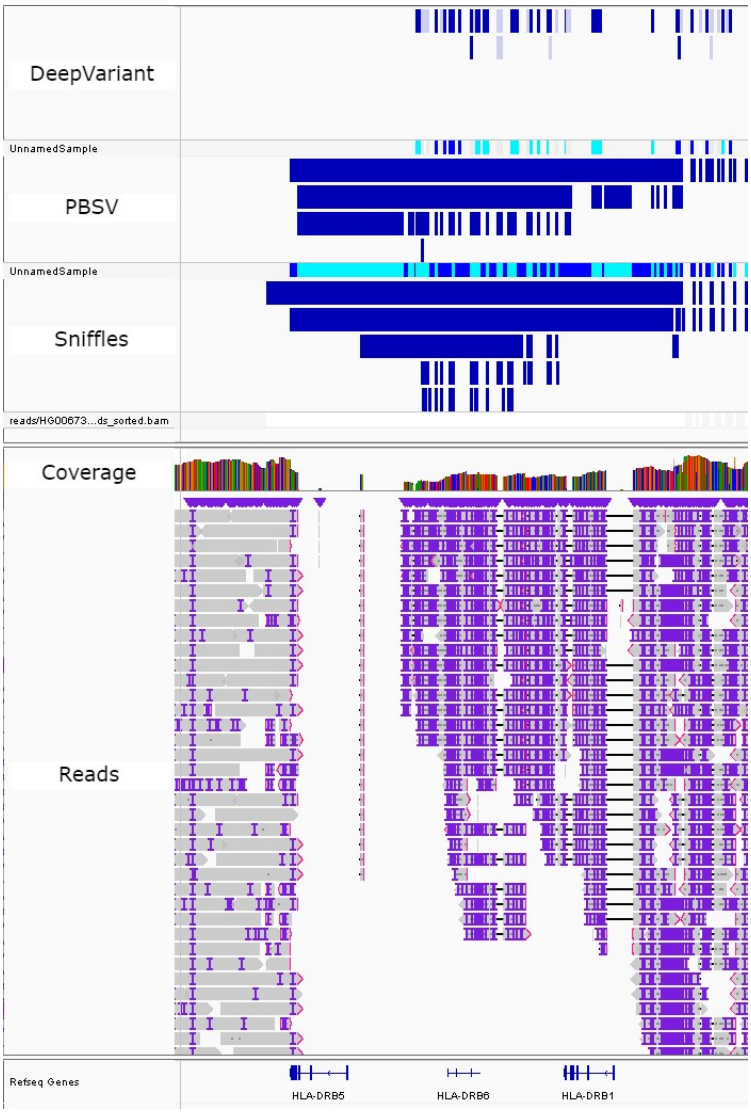
[21] "IGH immunoglobulin heavy locus [Homo sapiens (human)] - Gene - NCBI." https://www.ncbi.nlm.nih.gov/gene/3492 (accessed May 03, 2022).

[22] S. Kosugi, Y. Momozawa, X. Liu, C. Terao, M. Kubo, and Y. Kamatani, "Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing," *Genome Biol.*, vol. 20, no. 1, p. 117, Jun. 2019, doi: 10.1186/s13059-019-1720-5.

Appendix A

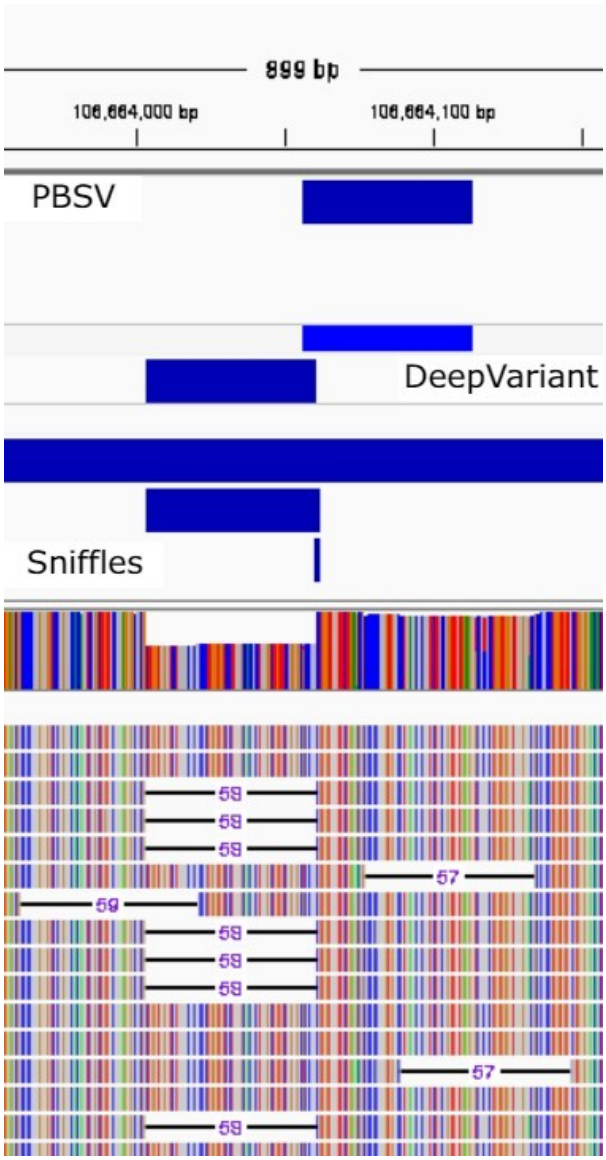
Code and data available at: https://github.com/MatusMikus/TUDelft-Thesis

Appendix A



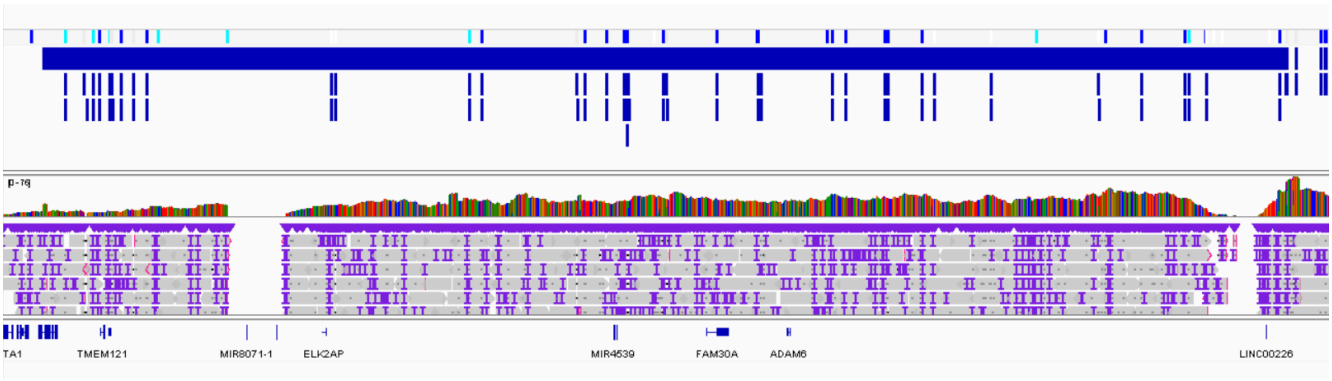
Differences in calling a difficult region with low coverage: PBSV and Sniffles call the whole region as a large insertion, DeepVariant calls individual variants within the insertion.

Appendix A

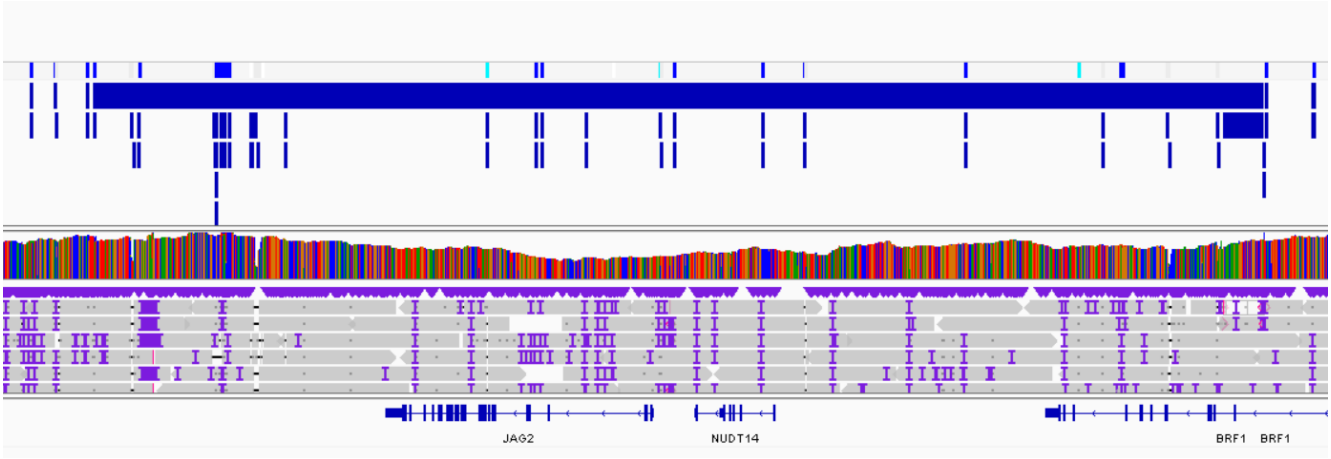


Same event, different ways of annotating. PBSV (on top) notices the same event from different reads, a likely consequence of read filtering to those that support a variant.

Appendix A



Sniffles Annotating an 800kbp inversion on chr14

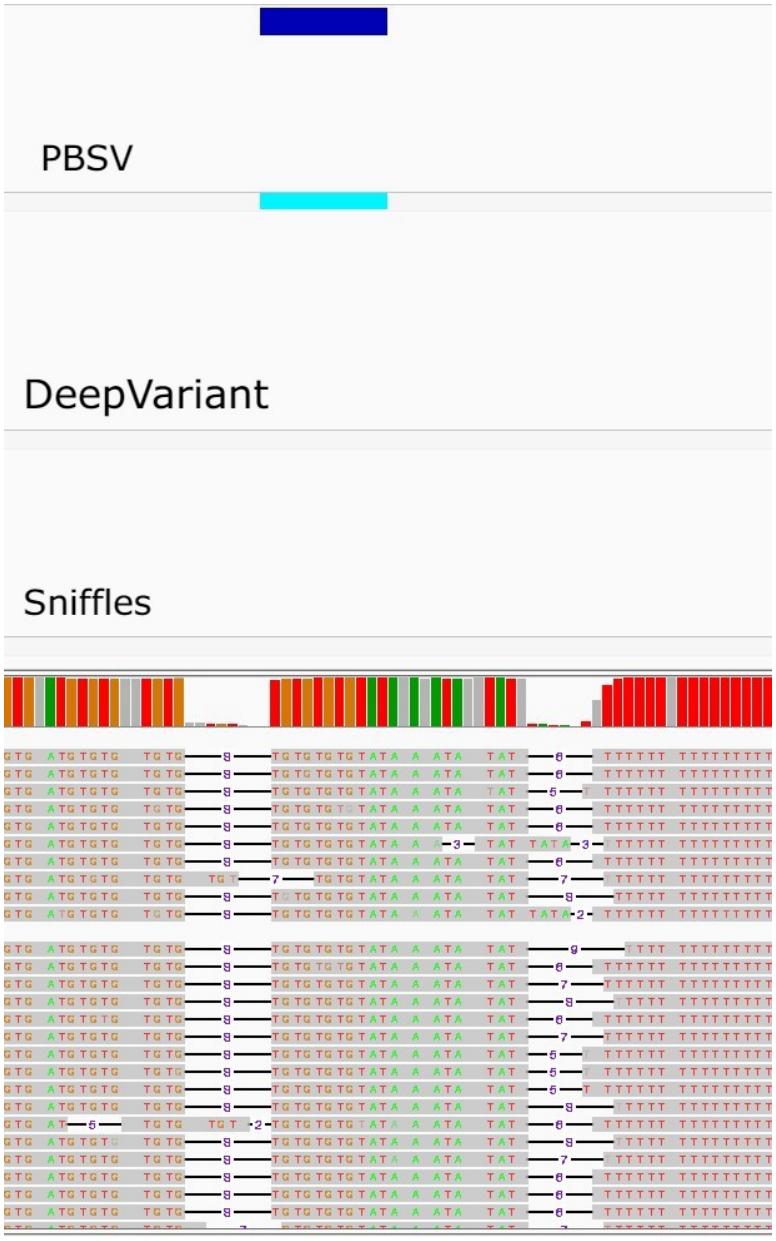


Sniffles annotating a 120kbp deletion on chr14, instead of noticing lower coverage

Appendix A

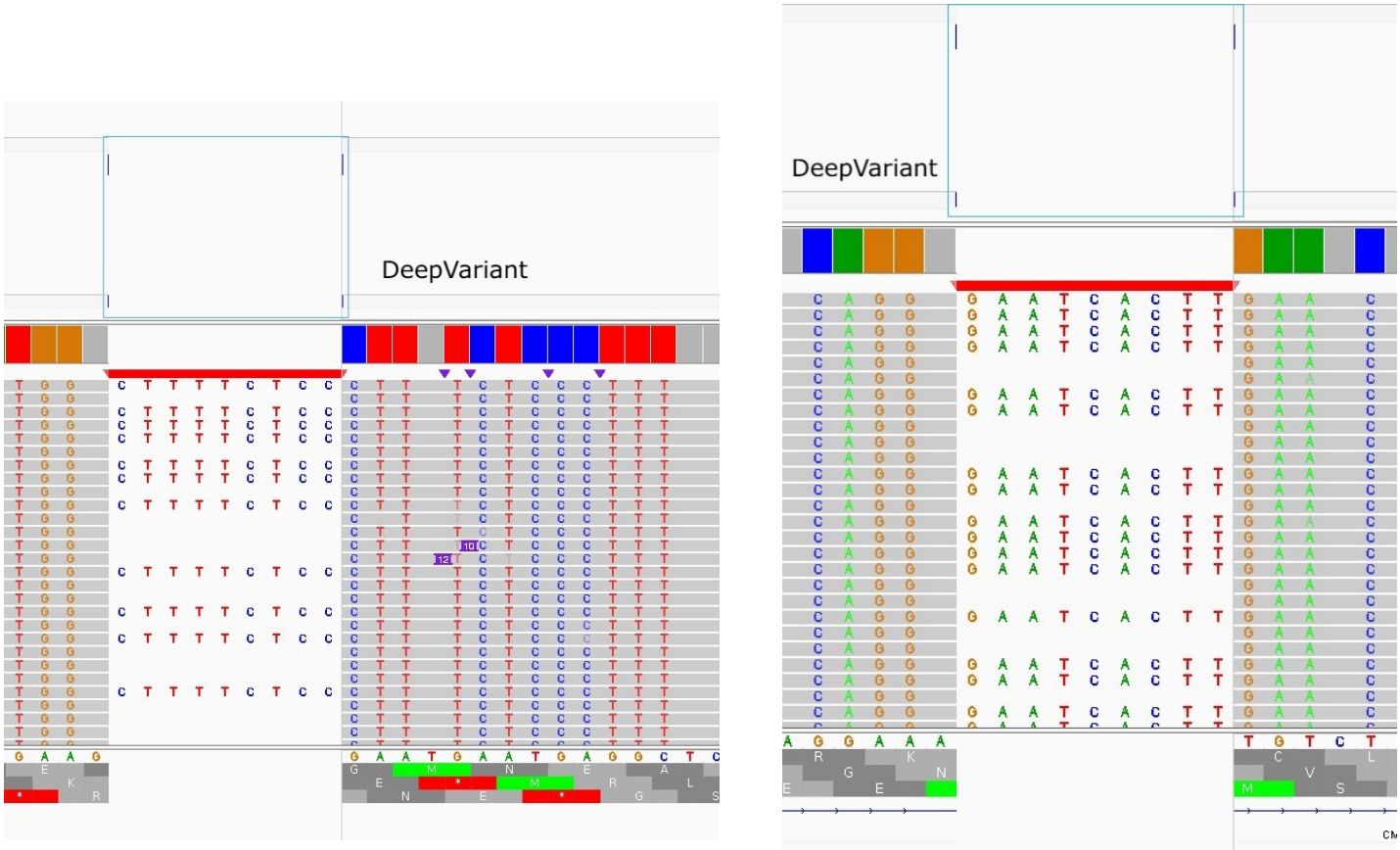
PBSV and Sniffles call this event as a variant, DeepVariant does not, perhaps due to a distribution of the insertions in the reads.

Appendix A



PBSV calling a variant that seems like a false positive and instead a read-generated error. The remaining two variant callers managed to avoid calling this as a variant

Appendix A



DeepVariant calling variants not called by the remaining callers. A duplication on the left, an insertion on the right