OXFORD

Briefings in Bioinformatics, 2024, **25(2)**, 1–10
https://doi.org/10.1093/bib/bbae049
Problem Solving Protocol

Kled: an ultra-fast and sensitive structural variant detection tool for long-read sequencing data

Zhendong Zhang †, Tao Jiang †, Gaoyang Li, Shuqi Cao, Yadong Liu, Bo Liu and Yadong Wang

Corresponding authors: Yadong Wang, Center for Bioinformatics, Faculty of Computing, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China. Tel.: +86-0451-86403963; Fax: +86-0451-86403963; E-mail: ydwang@hit.edu.cn; Yadong Liu, Center for Bioinformatics, Faculty of Computing, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China. Tel.: +86-0451-86403260; Fax: +86-0451-86403260; E-mail: ydliu@hit.edu.cn; Bo Liu, Center for Bioinformatics, Faculty of Computing, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China. Tel.: +86-0451-86413309; Fax: +86-0451-86413309; E-mail: bo.liu@hit.edu.cn

†Zhendong Zhang and Tao Jiang contributed equally to this work.

Abstract

Structural Variants (SVs) are a crucial type of genetic variant that can significantly impact phenotypes. Therefore, the identification of SVs is an essential part of modern genomic analysis. In this article, we present kled, an ultra-fast and sensitive SV caller for long-read sequencing data given the specially designed approach with a novel signature-merging algorithm, custom refinement strategies and a high-performance program structure. The evaluation results demonstrate that kled can achieve optimal SV calling compared to several state-of-the-art methods on simulated and real long-read data for different platforms and sequencing depths. Furthermore, kled excels at rapid SV calling and can efficiently utilize multiple Central Processing Unit (CPU) cores while maintaining low memory usage. The source code for kled can be obtained from https://github.com/CoREse/kled.

Keywords: structural variation; variant calling; long-read sequencing

INTRODUCTION

Genetic variation, which is a quintessential hereditary characteristic, has been definitively linked to critical human diseases and complex phenotypes. Single Nucleotide Variants (SNVs) and short insertions/deletions (indels) have been important genetic resources for uncovering the human population diversity and extensively used in genome-wide association studies [1–3]. Structural Variants (SVs), which are generally defined as variations that influence areas at a minimum of 50 base pairs (bp), are another type of variant. Compared to SNVs and indels, SVs have the largest numbers of altered base pairs [4–7] and more complex subtypes (e.g. deletions, insertions, duplications, inversion, translocation) [6–8], thus playing a significant role in genetic and clinical studies [9–13]. Therefore, the accurate and rapid detection of SVs is a fundamental and crucial task that facilitates further advancements in genomics.

The advancement of Next-Generation Sequencing (NGS) technology enables the cost-effective production of billions of short reads, which are typically paired and <1000 bp in length, for an individual [14] and can be used to discover various genetic variants using advanced variant callers [15–19]. However, the relatively short read length and systematic sequencing bias limit the accurate and sensitive detection of SVs, especially those larger than the read length or located in the repetitive regions of the genome [20]. As such, the human genome still harbors a vast amount of unexplored and unexcavated regions.

Recently, Third-Generation Sequencing (TGS) technologies such as Pacific Biosciences (PacBio) [21] and Oxford Nanopore Technologies (ONT) [22] can generate much longer sequencing reads (ranging from thousands to millions of base pairs), thereby greatly benefiting SV detection [23, 24], Telomere-to-Telomere (T2T) genome assembly [25], and human pan-genome construction [26, 27] among other applications. Numerous long-read-based SV detection methods [28–34] have been developed to achieve high-quality SV calling. For example, Sniffles [28, 29] identifies potential SV-supporting alignments based on statistics and clusters them based on various factors. CuteSV [30, 31] collects signatures from the aligned sequencing data and clusters them based on a defined distance metric. SVIM [32] collects signatures within and among alignments and performs graph clustering to identify maximum cliques as clusters. Using TGS-based approaches, over 20 000 SVs can be identified in a human individual, which is several times more than the number identified by NGS-based tools [35].

Zhendong Zhang is a PhD candidate in the Center for Bioinformatics at Harbin Institute of Technology. His research focuses on variation detection algorithms.
Tao Jiang is an associate professor in the Center for Bioinformatics at Harbin Institute of Technology. His research focuses on variation detection algorithms.
Shuqi Cao is a PhD candidate in the Center for Bioinformatics at Harbin Institute of Technology. Her research focuses on variation detection algorithms.
Yadong Liu is an assistant professor in the Center for Bioinformatics at Harbin Institute of Technology. His research focuses on genomic big data analysis.
Bo Liu is a professor in the Center for Bioinformatics at Harbin Institute of Technology. His research focuses on genome assembly algorithms.
Yadong Wang is a professor in the Center for Bioinformatics at Harbin Institute of Technology. His research focuses on developing bioinformatics methods and tools for next-generation and long-read sequencing data.
Received: September 15, 2023. **Revised:** January 12, 2024. **Accepted:** January 26, 2024

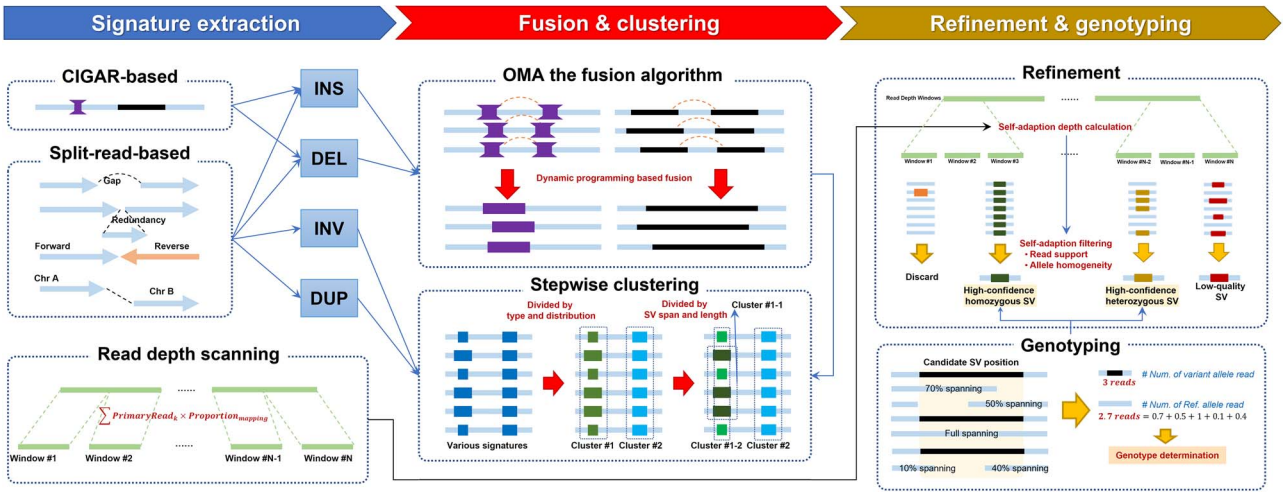

Figure 1. Schematic diagram of the overall design of kled. Step 1: Signatures are collected from CIGAR strings and split read information, and the read depth information is collected. Step 2: CIGAR signatures are first merged by an innovative algorithm called OMA, and the signatures are separated by SV types and areas and subsequently clustered by spans and lengths. Step 3: The clusters are refined by the supporting reads and lengths using self-adaptive criteria, after which a genotyping phase based on the supporting reads is conducted.

Although these TGS-based SV calling tools significantly advance high-quality SV discovery, two overlooked issues must be urgently addressed. First, the relatively high sequencing errors in the data generated from the PacBio Complete Long Reads (CLR) model and ONT (before the duplex model) are major challenges for current SV detection approaches [28–30, 32]. Achieving high performance on these error-prone data compared to the more accurate but expensive PacBio HiFi data would be beneficial for genome researchers. Second, there is a high demand for SV calling efficiency. With the development of large-scale population projects [6, 36–39], there is an urgent need for a more efficient and scalable detection approach to tackle the analysis challenges posed by the burgeoning long-read sequencing data.

To address these problems, we developed kled, an ultra-fast and sensitive SV detection tool with three specifically designed features: an innovative merging algorithm to mitigate the effect of the high sequencing errors, customized refinement strategies to enhance the calling performance, and a highly parallel and efficient program structure in C++, which is designed to significantly boost efficiency while maintaining low memory usage. Under robust experiments, kled yields superior SV results at a faster running speed than state-of-the-art methods across various sequencing technologies and depths. Moreover, kled delivers similar and outstanding SV calling performance on both error-prone dataset and HiFi dataset at a sequencing depth of 20× or higher. We believe that with its high speed and meticulous SV detection capabilities, kled holds great potential to usher in new opportunities and advancements in cutting-edge genomic research.

MATERIALS AND METHODS

Using sorted BAM file(s) as the input, kled extracts and analyzes the signatures to accomplish SV calling in three main steps (Figure 1).

Step 1: kled collects the SV signatures for different types of SVs, along with the read depth and primary alignment information.

Step 2: kled implements an innovative merging algorithm to further polish the signatures, followed by clustering the signatures using a method inspired by a previous work [34].

Step 3: kled refines and genotypes each generated signature cluster to generate and output the final VCF records (https://samtools.github.io/hts-specs/VCFv4.2.pdf).

Signature collection from CIGARs and split reads

Kled extracts two types of signatures: large insertions/deletions from CIGARs and split reads, which are consistent with most advanced methods. Kled also has two unique features as follows.

First, kled simultaneously records the start and end positions of all primary alignments into a dynamic array of intervals denoted as *PrimarySegments* during the signature collection from CIGARs; simultaneously, kled counts the read depth into pre-defined read depth windows, which are denoted by *RDWindows*, based on the following formula:

$$RDWindows_i = \sum_{OPA} \frac{Length\ of\ Overlap}{WindowSize}, \quad (1)$$

where $OPA \in \{Primary\ Alignments\ overlap\ [i * WindowSize, (i+1) * WindowSize)\}$, $WindowSize$ is pre-defined as 100; $RDWindows_i$ is the $i-th$ window, $i = 1, 2, \ldots, \left\lceil \frac{Length\ of\ the\ Chromosome}{WindowSize} \right\rceil$, and these *RDWindows* are defined for each chromosome. These collected data will be used in the refinement and genotyping steps, which will be illustrated in their sections.

Second, during the signature collection from the split reads, kled categorizes the inversion signatures into three types: the L type, R type and LR type, which correspond to clips with left, right or both orientations, respectively. This categorization helps kled reduce false positives in the refining phase.

Finally, an SV signature is generated that contains the template name from which it is originated, the beginning and ending positions, the supported SV type, the inserted DNA string, the L/R information of the inversion, and other auxiliary variables.

The problem of fragmented CIGAR entries

Before moving to the next step, a problem must be noticed. Due to the inherent high error rate in long-read sequencing data, the signature of deletion or insertion may be fragmented into smaller parts of the CIGAR entries during mapping. To reconstruct the signatures, current tools predominantly employ a straightforward approach by merging all successive CIGAR signatures whose distance is less than the pre-defined threshold (M) for a specified variant type [28–30]. However, this approach indiscriminately merges all CIGARs, regardless of whether they are actually fragments from the same variation or different nearby variations. Furthermore, even if this strategy correctly segregates the CIGARs

into two nearby signatures, the determination of the boundary lacks a convincing rationale.

Omni merging algorithm

To address the aforementioned issue, we introduce an innovative method to merge CIGAR signatures for deletions and insertions within the same alignment into larger deletion and insertion signatures using the information of nearby alignments, specifically, the Omni Merging Algorithm (OMA).

Given a list of ordered signatures for a specific type (deletion or insertion) from an alignment, we can construct a graph, where the nodes represent the signatures. Nodes that represent adjacent signatures can form an edge, and a presented edge signifies the merging of two adjacent signatures, which results in a new, merged signature. Consequently, a set of edges represents a merging strategy that can generate a new list of ordered signatures. The OMA is designed to identify the optimal merging strategy by assigning scores to different strategies and finding the strategy with the highest score. Specifically, the OMA uses signatures from nearby alignments to reinforce a specific merging strategy of the given alignment. If a signature generated by a merging strategy from another alignment resembles a signature generated by this strategy, it is considered supportive; in this case, it contributes to the score of this merging strategy.

The OMA is a search problem with exponential time complexity regarding the signature number. However, the component of finding supporting scores can be optimized using dynamic programming, and the number of merge-able edges can be constrained. These optimizations and other practical limitations make the OMA implementation highly computable. For a detailed description of the OMA, please refer to Supplementary Notes Part 1.

Clustering of SV signatures

When the signatures have been collected and reconstructed, kled separately clusters the signatures for different types of SVs. For a specific SV type, kled initially treats all corresponding signatures as independent single clusters and subsequently attempts to merge any two clusters C_1 and C_2 if there exist two sibling signatures S_1 and S_2 that satisfy $S_1 \in C_1$ and $S_2 \in C_2$. S_1 and S_2 are considered siblings if any condition in (2) and (3) is satisfied.

$$\max\left(\left|S_{1Left} - S_{2Left}\right|, \left|S_{1Right} - S_{2Right}\right|\right) < F, \tag{2}$$

$$\frac{\max\left(\left|S_{1Left} - S_{2Left}\right|, \left|S_{1Right} - S_{2Right}\right|, \left|S_{1Length} - S_{2Length}\right|\right)}{\min\left(S_{1Length}, S_{2Length}\right)} < CR, \tag{3}$$

where S_{iLeft} is the leftmost position of S_i, S_{iright} is the rightmost position of S_i, and $S_{ilength}$ is the length of S_i, $i = 1, 2$. For different SV types, the predetermined F and ratio CR can be specifically set.

Cluster refinement and SV calling

All clusters generated in the last step are considered SV candidates, the average position and length of the signatures in the cluster are determined as the SV position and length. To accurately report the SVs, kled applies specially designed filtering strategies to filter out potential false positives. Specifically, kled establishes two criteria to proceed with the filtering, each with a threshold for *ST* and *LS*, where *ST* is the number of supporting reads, and *LS* is defined as follows:

$$LS = \max\left(0, 100 - \left(\frac{LSD}{SVLen}\right) * 100\right), \tag{4}$$

where *SVLen* is the SV length, and *LSD* is the standard deviation of the lengths of the signatures in the cluster. If a cluster fails both criteria, it is deemed to have failed the filters. The thresholds of *ST* are determined based on the SV type and sequencing depths, which are calculated using the aforementioned read depth data. Kled sets *ASSBase* and *ASSRatio* for each SV type and uses $CV \cdot ASSRatio + ASSBase$ as the threshold for the number of supporting reads. This approach eliminates the need for manual input of thresholds. Since the read depth data are collected during the signature collection step, this mechanism does not introduce extra input/output (I/O) costs.

Moreover, kled applies extra refinement conditions for specific SV types. As explained in the signature collection section, inversion signatures can be classified into three types: L, R and LR types. If a cluster has only L-type or R-type signatures, it is filtered out. Furthermore, if a cluster has more supporting signatures than twice the number of supporting reads for insertions and duplications, it is filtered out, since it is more likely to be a false discovery. If a chromosome has a high *DuplicationRatio*, where $DuplicationRatio = \frac{Number\ of\ Duplications}{Number\ of\ Insertions}$, kled filters out the insertion clusters in this chromosome if they have high position standard deviations and at least one nearby duplication cluster to reduce the number of false positives that report duplications as insertions.

Genotyping

An SV candidate, after passing the refinement step, is reported as an SV record in the final VCF file with the determination of the zygosity. Kled compares the average coverage of the region (*AC*) to the number of supporting reads (*ST*) to determine whether the SV is heterozygous or homozygous. If *ST*/*AC* is less than a certain threshold (default 0.8), the genotype is considered to be '0/1' and otherwise '1/1.' For insertions, instead of comparing *ST* to *AC*, kled compares *ST* to the number of reads whose primary alignments contain the local region of the insertion breaks (*AT*). If *ST*/*AT* is less than a certain threshold (default 0.75), the genotype is considered '0/1'; otherwise, it is considered '1/1.' *AC* and *AT* are calculated using the read depth and primary segments counted during the signature collection, which eliminates the need for I/O in this phase and greatly improves the speed of the process.

Implementation of a high-performance structure

We designed kled to rapidly detect SVs with a focus on both methodology and program structure. In terms of methodology, kled records the read depth and primary alignment information during the signature collection, which eliminates the need to re-scan the alignment files during the refinement and genotyping, which significantly enhances the speed. Although the use of dynamic programming reduces the time complexity in the OMA, substantial computational power is still necessary. Kled addresses this issue by limiting the merge-able edges and subsampling the supporting alignments to a limited number, which reduces time complexity.

For the program structure, the entire program is written in C++ to ensure superior running efficiency. Furthermore, kled utilizes high-performance libraries. For example, during the signature collection phase, kled employs the multi-threading I/O function of htslib (https://github.com/samtools/htslib), which significantly enhances the I/O speed. In addition to the I/O part, kled incorporates multi-threading schedules in most sections. Within the OMA, in conjunction with the methodological designs, the multi-threading structure enables kled to swiftly complete the merging

process. Moreover, kled divides the calling procedures after signature collection by chromosomes and SV types and distributes them into the threading pool, which enables them to be executed in parallel. Regarding memory usage, kled ensures that most data only hold a single copy throughout multi-threading, reducing memory consumption.

Implementation of simulation and SV calling

We simulated long-read data for benchmarking. Specifically, we used VISOR [40] v1.1.2 to add the randomly simulated 24,919 SVs, including 12,365 deletions, 12,176 insertions, 185 tandem duplications (hereafter denoted as duplications for brevity), and 193 inversions, all of varying SV sizes and zygosities, along with the SNVs of HG00403 and randomly simulated small indels (<50 bp), to a GRCh38 reference, to obtain two haplotypes. Using this modified genome as the donor, we employed lrsim (https://github.com/CoREse/lrsim) v0.2 to simulate 30× long-read data. Subsequently, Minimap2 [41] v2.17-r941 was used for long-read mapping, and SAMtools [42] v1.19 was used for data down-sampling (i.e. 5×, 10× and 20×). All scripts are available at https://github.com/CoREse/kled.

To fully evaluate the performance of kled (v1.2.9) and three state-of-the-art SV calling tools for comparison [cuteSV (git commit 9ca0f93), Sniffles2 (v2.0.7) and SVIM (v2.0.0)], we conducted experiments on the simulated data and real data from PacBio HiFi, PacBio CLR and ONT platforms. After SVIM generated the VCF file, we used bcftools v1.10.2 to sort the VCF file and filter the results according to the number of supporting reads; we used the same value that cuteSV used for the minimum number of supporting reads because SVIM outputs all SV candidates without filtering. Subsequently, we used in-house scripts to add tags including SVLEN and SVTYPE to some records in the SVIM VCF file. Please refer to Supplementary Notes Part 2 for more details.

Benchmarking of the SV detection performance

We selected Truvari (https://github.com/ACEnglish/truvari) v2.1 to evaluate the performance of each tool in SV calling using the parameters '*-r 1000 -p 0.00 –passonly*.' For the simulated data, we used the generated variant file as the ground truth set; this was also used to generate the simulated donor genome. For the HG002 human sample data, we used the Tier 1 SV benchmark set for HG002 (v0.6) released by the Genome In A Bottom consortium (GIAB) [43] as the ground truth. The precision, recall, and F1 score calculated from Truvari were used to describe the performance of the SV detection, which are defined as follows:

$$Precision = \frac{TP_{call}}{TP_{call} + FP_{call}}, \tag{5}$$

$$Recall = \frac{TP_{base}}{TP_{base} + FN_{base}}, \tag{6}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{7}$$

where TP_{call} and FP_{call} are the number of correct and incorrect predictions, respectively; TP_{base} and FN_{base} are the number of correctly covered and not covered records in the ground truth set, respectively.

In the assessment of Mendelian Discordance Rate (MDR), we used Jasmine (https://github.com/mkirsche/Jasmine) v1.1.1 to integrate the SV results produced by each SV caller on the GIAB Ashkenazi trio [43] samples (i.e. HG002, HG003 and HG004). Then, we collected all SV records that disobeyed the laws of heredity to calculate the MDR:

$$MDR = \frac{\sum SV_{discordant}}{\sum SV_{consistent}}, \tag{8}$$

where $SV_{discordant}$ represents SVs that occurred in all three samples and did not satisfy Mendel's laws between the child and his parents, and $SV_{consistent}$ represents SVs that occurred in all three samples. Please refer to Supplementary Notes Part 2 for more details.

Running time and memory usage

To assess the speed and memory consumption performance of kled, we tested four SV detection tools using 30× PacBio HiFi, PacBio CLR and ONT data with thread counts of 1, 2, 4, 8 and 16 (only 1 thread was tested for SVIM because it only supports single-threading). We used the GNU time for time recording and tmemc (https://github.com/CoREse/tmemc) for memory usage recording. Tmemc utilizes smem (https://www.selenic.com/smem/) to record the maximum total physical memory usage of the process and its subprocesses (if any). The tests were conducted on a platform with an AMD Ryzen 9 3950X, 128GB of DDR4 RAM dual channel at 4800 MHz, and Ubuntu 22.04.3 LTS. You may refer to Supplementary Notes Part 2 for more details.

RESULTS

Results on simulated datasets

First, we generated simulation data to assess the performance of kled as a baseline. Here, we applied VISOR to generate a modified donor genome and subsequently produced 30× error-prone long-read sequencing data using lrsim. Three state-of-the-art SV calling methods were used for benchmarking: cuteSV, Sniffles2 and SVIM. In total, kled achieved the best SV calling performance (F1: 95.45% for presence and F1: 94.65% for genotype) and outperformed the runner-up method, Sniffles2, by 0.45 and 0.98%, respectively (Figure 2, Supplementary Table S1). For most SV types, kled obtained the best performance for SV presence (F1: 95.39% for deletion, F1: 79.61% for duplication and F1: 93.77% for inversion), which are 0.06–17.16% higher than other approaches. In terms of insertions, kled maintained competitive performance and achieved an F1 that was only 0.04% lower than that of Sniffles2. When further considering the genotypes, kled obtained the best performance for all SV types (F1: 94.85% for deletion, 94.86% for insertion, 65.22% for duplication and 89.89% for inversion), with their F1 values all being 0.02–16.86% higher than those of other approaches (Figure 2, Supplementary Tables S2–S5). SVIM did not provide genotyping information about the tandem duplications.

To benchmark the performance of these tools on data with lower sequencing depth, we randomly down-sampled the 30× dataset using SAMtools to obtain 5×, 10× and 20× sequencing datasets. For these datasets, kled continued to achieve the best SV calling performance in terms of SV presence (F1: 82.07% on 5× data, F1: 92.46% on 10× data, and F1: 95% on 20× data), with leads of 0.22–4.13% compared to other tools. For genotyping, kled maintained the best performance on 10× and 20× data (GT-F1: 88.86% on 10× data and GT-F1:93.67% on 20× data), with advantages of 0.53–4.41% compared to other tools. For the relatively low 5× data, kled yielded a 73.8% GT-F1 score, so it remains in the top tier, is 0.43% lower than Sniffles2, and outperforms the other two tools by 2.38 and 17.89% (Figure 2, Supplementary Table S1). The results of simulated datasets demonstrate that kled can achieve

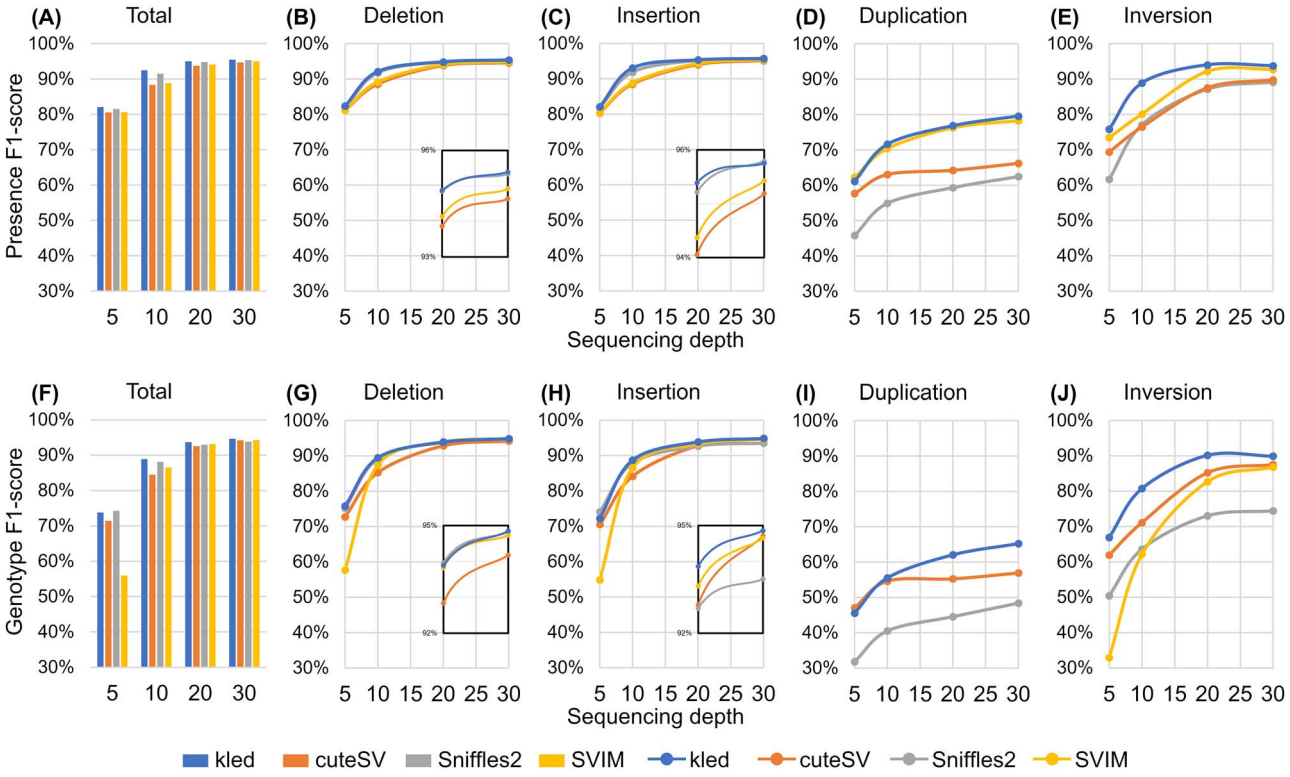

Figure 2. F1 score comparisons on simulated data. The vertical axes denote the F1 scores for presence or genotype; the horizontal axes represent the sequencing depths of the data. The sub-figures include (**A**) Overall presence of F1 score benchmarking; Presence of F1 score benchmarking for (**B**) deletions (with an enlarged view for 20× and 30× sequencing depths), (**C**) insertions (with an enlarged view for 20× and 30× sequencing depths), (**D**) duplications and (**E**) inversions. (**F**) Overall genotype F1 score benchmarking; Genotype F1 score benchmarking for (**G**) deletions (with an enlarged view for 20× and 30× sequencing depths), (**H**) insertions (with an enlarged view for 20× and 30× sequencing depths), (**I**) duplications and (**J**) inversions.

SV calling in high sensitivity and precision, and it has stable performance even at low coverage.

Results on real datasets of the HG002 individual

Simulated data demonstrate the superior SV detection performance of kled, but real-world sequencing data must be assessed since they are more complex and comprehensive. Here, we selected the well-studied human sample HG002 and collected three 30× sequencing data about it, which were generated on the platforms of PacBio HiFi, PacBio CLR and ONT. The GIAB Tier 1 SV benchmark set for HG002 (v0.6, containing 4199 and 5442 high-confidence deletions and insertions) was used as the ground truth dataset. Overall, kled produced the best results for SV presence on PacBio CLR and ONT data (F1: 94.22% for presence and 92.16% for genotype on PacBio CLR data; F1: 94.64% for presence and 93.29% for genotype on ONT data), where the minimal leads are 1.06 and 0.83% leads compared to other tools for presence F1 and genotype F1, respectively. For PacBio HiFi data, kled obtained the highest genotype F1 score (94.02%) and a competitive presence F1 score (94.88%), which is only 0.06% lower than cuteSV and at least 0.97% higher than the other tools (Figure 3, Supplementary Table S6). In terms of the performance on each SV type, kled achieved the highest F1 scores for both presence and genotype for insertion across all platforms and the best or second-best F1 scores for presence or genotype for deletion on different platforms (Figure 3, Supplementary Tables S7 and S8).

Similarly, we benchmarked these tools on diverse low-depth sequencing data. At a sequencing depth of 5×, kled maintained the best results (F1: 79.33%; GT-F1: 72.15%) on ONT platforms and remained competitive on the PacBio HiFi and CLR platforms. For 10× and 20× data, kled showed the best performance (F1: 92.76, 82.07 and 89.81%, and GT-F1: 89.79, 76.53 and 85.78% on 10× data; F1: 94.61, 91.69 and 93.47% and GT-F1: 93.38, 88.84 and 91.5% on 20× data, for PacBio HiFi, PacBio CLR and ONT platforms, respectively); there is a maximum 6.9% advantage over the next best tool. The sequencing depth of 10× is a turning point for kled, after which, for all three sequencing platforms, kled yielded a minimum 80% F1 score, which indicates a great potential for kled to be used in cost-efficient sequencing tasks (Figure 3, Supplementary Table S6).

Kled exhibits significant advantages when processing data from error-prone platforms, including PacBio CLR and ONT, with considerable improvements on 30× data compared to the other tested tools (1.06–11.75% and 1.69–6.94% for presence F1 leads and 0.83–11.46% and 1.53–11.42% for genotype F1 leads on PacBio CLR and ONT data, respectively; Figure 3, Supplementary Table S6). The advantages of kled in processing error-prone datasets are mainly obtained by the innovative CIGAR merging algorithm OMA (Materials and methods, Supplementary Notes Part 1), which is designed to minimize the impact of sequencing and mapping errors. Although PacBio HiFi data will yield better SV detection results than other platforms due to its lower error rate, kled achieved comparable results to PacBio HiFi data for PacBio CLR and ONT data when the depth reached 20× (Figure 3, Supplementary Tables S6–S8), which is also attributed to the OMA and highlights its significant potential for long-read sequencing analysis. In conclusion, the experiments conducted on real data demonstrate the capabilities of kled to accomplish

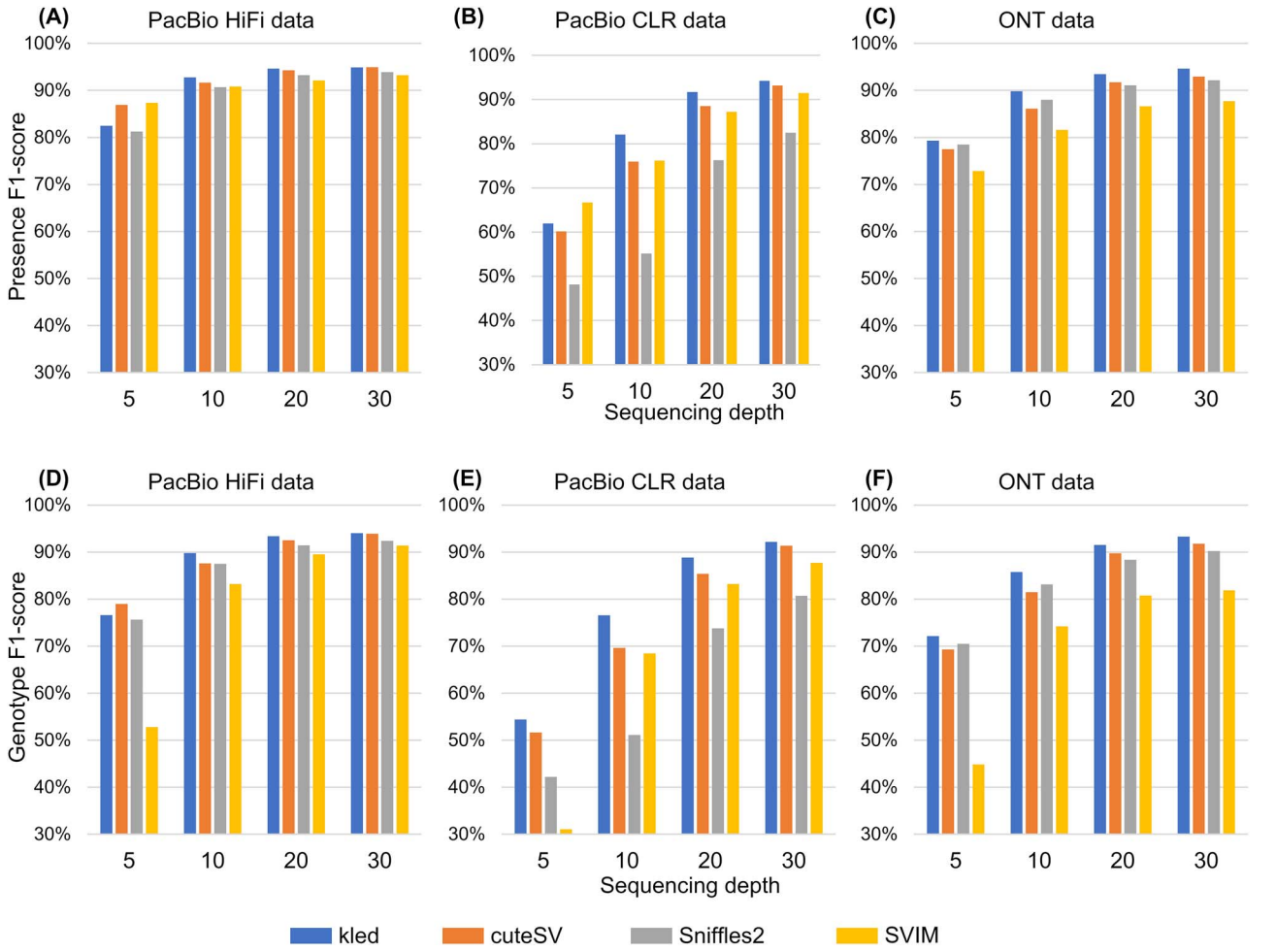

Figure 3. Benchmarking results on the Genome in a Bottle consortium (GIAB) SV dataset for HG002. The vertical axes denote the F1 scores or genotype F1 scores, and the horizontal axes represent the sequencing depths of the data. The sub-figures include presence F1 score benchmarking on (**A**) PacBio HiFi data, (**B**) PacBio CLR data, and (**C**) ONT data; genotype F1 score benchmarking on (**D**) PacBio HiFi data, (**E**) PacBio CLR data and (**F**) ONT data.

the SV detection task using real data from all three major TGS platforms. In addition, kled proves to be superior to state-of-the-art tools, particularly for data from error-prone platforms.

Mendelian discordance rate test on the Ashkenazi trio data

Mendel's law of inheritance is typically one of the authoritative methods to measure the variant calling performance. Hence, we selected the GIAB Ashkenazi trio datasets [43] (i.e. 52× HG002 ONT data, 93× HG003 ONT data, 90× HG004 ONT data) to further evaluate the MDR for consistent SV loci. Specifically, we utilized each tool to complete the SV calling on this trio and subsequently calculated the MDR using our in-house scripts (Materials and methods, Supplementary Notes Part 2). In general, kled achieved the lowest overall MDRs among the four tested tools (i.e. 0.91% compared to 1.31–1.66% achieved by other tools) with a large number of consistent SV loci (24446); kled only has fewer consistent SV loci than cuteSV (Figure 4, Supplementary Table S9). For each SV type assessment, kled also achieved excellent MDR performances with large numbers of consistent SVs, which illustrates the great self-consistency of kled (Figure 4, Supplementary Table S9). Specifically, for insertions, duplications and inversions, kled obtained the lowest MDRs (0.95, 0 and 0%, respectively), which are at most 1.85% lower than the MDRs achieved by the other three tools. Kled yielded the highest numbers of consistent SV loci for duplications (343) and inversions (82). The good MDR performance of kled is mainly due to its dedicatedly designed refinement mechanisms and accurately counted read depths, which are used by the genotyping phase and make the outputs more precise.

Tests of the running speed and memory consumption

The above experiments demonstrate the outstanding ability of kled to comprehensively and accurately detect SVs. With the explosive growth of sequencing data, there is a need for precise SV detection and a high demand for efficiency. To assess the runtime and memory footprints for these methods, we conducted tests utilizing 1, 2, 4, 8 and 16 thread(s) on the PacBio HiFi, PacBio CLR and ONT data. In terms of runtime, the benchmarking results confirmed that kled unequivocally achieved the fastest speed, which is at least double the speed of other tools (e.g. 2.74–15.08 times as fast in the single-threading mode and 2.06–6.25 times as fast using 16 threads across all platforms, as shown in Figure 5 and Supplementary Table S10). In the single-threading mode on PacBio CLR data, kled even achieved a speed that was 5.64–15.08 as fast as that of other tools (Supplementary Table S10). Among the tested software tools, kled, cuteSV and Sniffles2 can utilize multiple cores for SV calling, whereas SVIM only supports the single-threaded mode. When the thread count increased from 1

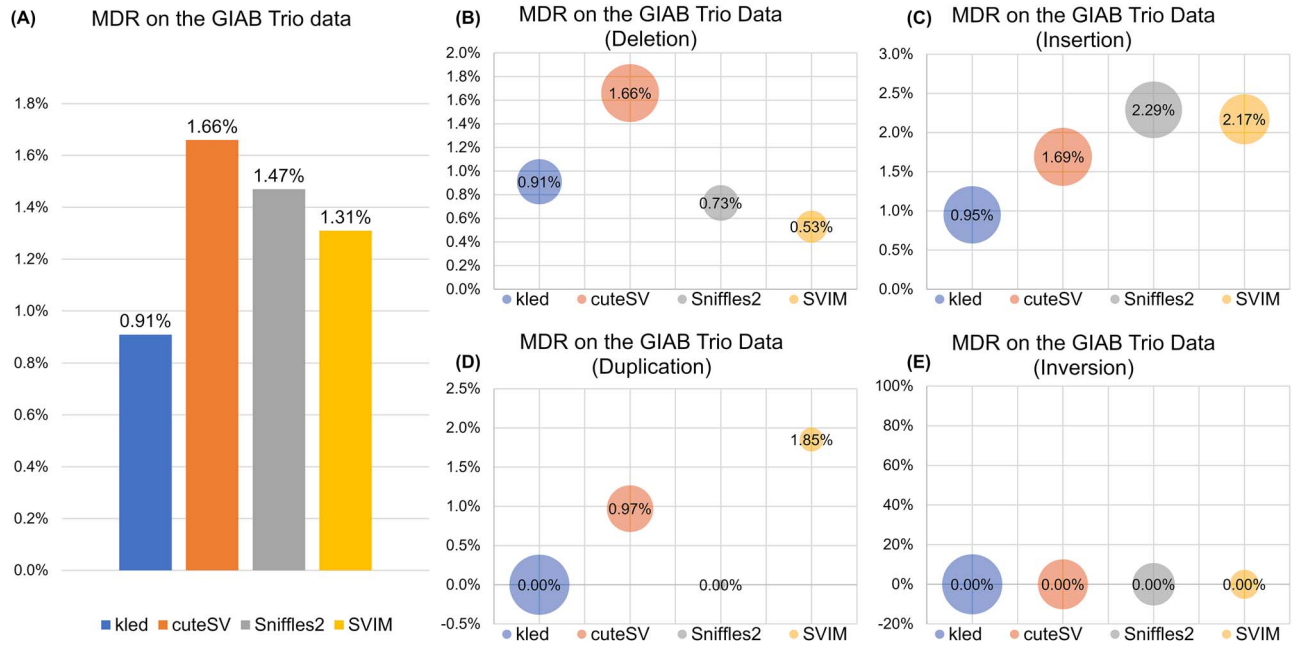

Figure 4. MDR comparisons on Ashkenazi trio family data. The vertical axes represent the MDR value, where lower values are preferable; the horizontal axes and colors mark different tools. In the SV-type specific diagrams, the sizes of the bubbles indicate the counts of consistent SV loci, and larger bubbles are more desirable. The sub-figures include (**A**) Histogram of the MDR values; MDR values and consistent SV loci counts for (**B**) deletions, (**C**) insertions, (**D**) duplications and (**E**) inversions.

to 16, kled exhibited a substantial speed enhancement, which highlights its efficiency in utilizing multiple cores (Figure 5, Supplementary Table S10). Specifically, from 1 to 8 threads, kled showed a near-linear speed increase; from 8 to 16 threads, the speed increases were less pronounced because the I/O part of kled reached a limitation. The superior efficiency of kled can be largely attributed to the high-performance program structure implemented in kled (Materials and methods). For memory usage, kled maintained efficient and stable memory usage across different data and thread counts, whereas Sniffles2 showed a significant increase in memory consumption when the thread count increased (Figure 5, Supplementary Table S11). Specifically, the experiments show that kled required <5 GB of physical memory to complete the SV detections on 30× data from PacBio HiFi, PacBio CLR or ONT platforms, so it is suitable for most modern personal computers. The superior speed performance and the low and stable memory usage indicate that kled is suitable for various research or practical tasks, from low-cost sequencing projects with limited computational resources to large-scale genome research projects that generate substantial data to be processed.

DISCUSSION AND CONCLUSION

In this article, we introduce kled, an ultra-fast and sensitive SV detection tool for long-read. Kled swiftly delivers optimal SV detection results due to its unique features including the innovative merging algorithm OMA, dedicated refining metrics and high-performance program structure. Specifically, kled has three main advantages compared to state-of-the-art SV detection tools.

First, kled is suited for high-quality SV calling tasks, especially those containing error-prone long reads. The reason is the specially designed signature-merging algorithm OMA, which can quickly collect the related signatures overall, generate accurate merged signatures in each alignment, and minimize the effects of sequencing errors and alignment bias. Here, two examples were provided, and they were correctly reported only by kled (Supplementary Figures S1 and S2). Kled is successful in these examples mainly because the OMA can merge long-spanning and more disorderly signatures.

Second, kled can achieve SV calling on a 30× human sample long-read sequencing data within only ~3 min using eight threads, which is at least one to seven times faster than the cutting-edge methods. This ultra-fast speed benefits from the designed parallel architecture, high-efficiency implementation in C++ and the application of high-performance third-party libraries. In addition, we optimized the memory usage so that kled can maintain a low and stable memory footprint. Based on these strengths, kled is suitable for tasks with high demand for efficiency and throughput.

Third, we established a self-adaptive threshold calculation mechanism for users to operate without having to determine the parameters themselves. Kled automatically calculates the actual read depth CV and determines the cutoff for the number of supporting reads. The criteria calculated based on the read depth may be more adaptable than manually set fixed parameters and can eliminate the possibility of manual errors. Thus, kled is accessible to everyone from ordinary users with no field knowledge to bioinformatics experts.

These specially designed algorithms and strategies enhance the ability to competently perform ultra-fast and sensitive SV calling on long-read data. However, several inherent issues in kled must be expounded to rationally employ kled in the appropriate scenarios.

Notably, kled is currently optimized specifically for the diploid genome. Although kled can detect SVs in species with different ploidies, the genotyping is not tailored for them, which leads to potentially undefined results. Because the ploidy of organisms can significantly vary across species and different ploidies can drastically alter the genotyping model, adapting genotyping to

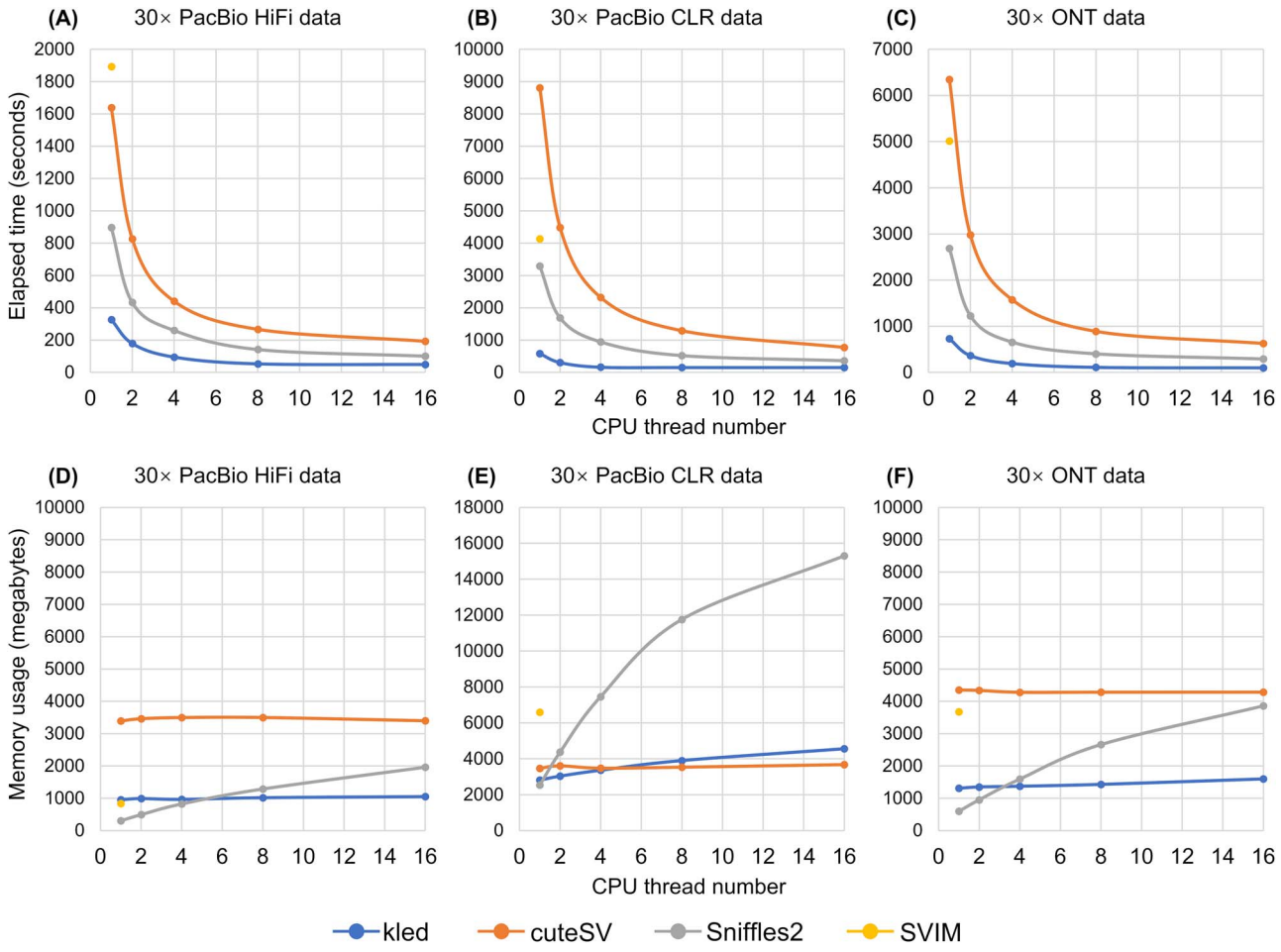

Figure 5. Line charts of the time (s) and maximum physical memory (MB) consumed to complete SV detections on 30× data from PacBio HiFi, PacBio CLR and ONT platforms. The vertical axes represent the time consumed or maximum physical memory occupied; the horizontal axes denote the thread counts of the detections. Sub-figures: Time consumptions on (**A**) 30× PacBio HiFi data, (**B**) 30× PacBio CLR data and (**C**) 30× ONT data; maximum physical memory requirements on (**D**) 30× PacBio HiFi data, (**E**) 30× PacBio CLR data and (**F**) 30× ONT data.

accommodate most ploidies is a formidable challenge for the current kled. Furthermore, as an alignment-based SV detection tool, kled may have disadvantages compared to de-novo assembly based methods because de-novo assembly (especially for the T2T genome) can provide a more complete and accurate colinear alignment to discover more hard-to-call SVs in complex genomic regions (e.g. segmental duplications, centromeres, telomeres, etc.). Additionally, kled, as well as the three state-of-the-art tools tested in the experiments, can detect duplications based purely on the split-read signatures. However, when a duplication is short, mapping tools tend to map it as a simple insertion in the CIGAR string; therefore, there is no split-read, resulting in a low recall for duplication identifications. This could be improved by introducing additional efforts, such as local realignments for all insertion signatures, which is potentially computationally costly.

With the high speed and meticulous SV detection capabilities of kled, kled will bring great potential to usher in new opportunities and advancements in cutting-edge genomic research.

Key Points

- Kled is an ultra-fast and sensitive structural variant (SV) calling approach for long reads and outperforms state-of-the-art SV detection tools.
- Kled specially designs a novel signature-merging algorithm and refinement strategies to better organize the signature clustering and processing.
- Kled is particularly good at detecting SVs on sequencing data from error-prone platforms such as PacBio CLR and ONT due to its novel signature-merging algorithm.
- Kled provides a self-adaptive threshold calculation mechanism and is accessible to users without domain knowledge.

SUPPLEMENTARY DATA

Supplementary data are available online at http://bib.oxfordjournals.org/.

FUNDING

This work was supported by National Natural Science Foundation of China (No: 62331012, 32000467), Natural Science Foundation of Heilongjiang Province (No: LH2023F014), China Postdoctoral Science Foundation (Nos: 2022 M720965 and 2020 M681086) and Heilongjiang Provincial Postdoctoral Science Foundation (No: LBH-Z20014).

DATA AVAILABILITY

The crucial parameters for each detection tool used in the experiments are listed in Supplementary Table S12. The reference genome, sequencing data and ground truth callsets in this study are listed in Supplementary Table S13. Versions of the software we used are listed in Supplementary Table S14. The implementation of kled can be accessed from GitHub (https://github.com/CoREse/kled).

REFERENCES

1. Kim S, Misra A. SNP genotyping: technologies and biomedical applications. *Annu Rev Biomed Eng* 2007;**9**:289–320.
2. Auton A, Abecasis GR, Altshuler DM, *et al.* A global reference for human genetic variation. *Nature* 2015;**526**:68–74.
3. Bennett EP, Petersen BL, Johansen IE, *et al.* INDEL detection, the 'Achilles heel' of precise genome editing: a survey of methods for accurate profiling of gene editing induced indels. *Nucleic Acids Res* 2020;**48**:11958–81.
4. Conrad DF, Pinto D, Redon R, *et al.* Origins and functional impact of copy number variation in the human genome. *Nature* 2010;**464**:704–12.
5. Kidd JM, Graves T, Newman TL, *et al.* A human genome structural variation sequencing resource reveals insights into mutational mechanisms. *Cell* 2010;**143**:837–47.
6. Sudmant PH, Rausch T, Gardner EJ, *et al.* An integrated map of structural variation in 2,504 human genomes. *Nature* 2015;**526**:75–81.
7. Ahsan MU, Liu Q, Perdomo JE, *et al.* A survey of algorithms for the detection of genomic structural variants from long-read sequencing data. *Nat Methods* 2023;**20**:1143–58.
8. Alkan C, Coe BP, Eichler EE. Genome structural variation discovery and genotyping. *Nat Rev Genet* 2011;**12**:363–76.
9. Weischenfeldt J, Symmons O, Spitz F, Korbel JO. Phenotypic impact of genomic structural variation: insights from and for human disease. *Nat Rev Genet* 2013;**14**:125–38.
10. Macintyre G, Ylstra B, Brenton JD. Sequencing structural variants in cancer for precision therapeutics. *Trends Genet* 2016;**32**:530–42.
11. Dennenmoser S, Sedlazeck FJ, Iwaszkiewicz E, *et al.* Copy number increases of transposable elements and protein-coding genes in an invasive fish of hybrid origin. *Mol Ecol* 2017;**26**:4712–24.
12. Chiang C, Scott AJ, Davis JR, *et al.* The impact of structural variation on human gene expression. *Nat Genet* 2017;**49**:692–9.
13. Jeffares DC, Jolly C, Hoti M, *et al.* Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast. *Nat Commun* 2017;**8**:14061.
14. Hu T, Chitnis N, Monos D, Dinh A. Next-generation sequencing technologies: an overview. *Hum Immunol* 2021;**82**:801–11.
15. Zhang Z, Liu Y, Li G, *et al.* PocaCNV: a tool to detect copy number variants from population-scale genome sequencing data. *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* 2021; 1912–8
16. Li G, Jiang T, Li J, Wang Y. PanSVR: Pan-genome augmented short read realignment for sensitive detection of structural variations. *Front Genet* 2021;**12**:731515.
17. Liu Y, Jiang T, Yan G, *et al.* Psi-caller: a lightweight short read-based variant caller with high speed and accuracy. *Front Cell Develop Biol* 2021;**9**:731424.
18. Chen X, Schulz-Trieglaff O, Shaw R, *et al.* Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics* 2016;**32**:1220–2.
19. Layer RM, Chiang C, Quinlan AR, Hall IM. LUMPY: a probabilistic framework for structural variant discovery. *Genome Biol* 2014;**15**:R84.
20. English AC, Salerno WJ, Hampton OA, *et al.* Assessing structural variation in a personal genome—towards a human reference diploid genome. *BMC Genomics* 2015;**16**:286.
21. Roberts RJ, Carneiro MO, Schatz MC. The advantages of SMRT sequencing. *Genome Biol* 2013;**14**:405.
22. Jain M, Olsen HE, Paten B, Akeson M. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. *Genome Biol* 2016;**17**:239.
23. Sedlazeck FJ, Lee H, Darby CA, Schatz MC. Piercing the dark matter: bioinformatics of long-range sequencing and mapping. *Nat Rev Genet* 2018;**19**:329–46.
24. Goodwin S, McPherson JD, McCombie WR. Coming of age: ten years of next-generation sequencing technologies. *Nat Rev Genet* 2016;**17**:333–51.
25. Nurk S, Koren S, Rhie A, *et al.* The complete sequence of a human genome. *Science* 2022;**376**:44–53.
26. Miga KH, Wang T. The need for a human Pangenome reference sequence. *Annu Rev Genomics Hum Genet* 2021;**22**:81–102.
27. Wang T, Antonacci-Fulton L, Howe K, *et al.* The human Pangenome project: a global resource to map genomic diversity. *Nature* 2022;**604**:437–46.
28. Sedlazeck FJ, Rescheneder P, Smolka M, *et al.* Accurate detection of complex structural variations using single-molecule sequencing. *Nat Methods* 2018;**15**:461–8.
29. Smolka M, Paulin LF, Grochowski CM, *et al.* Detection of mosaic and population-level structural variants with Sniffles2. *Nat Biotechnol* 2024.
30. Jiang T, Liu Y, Jiang Y, *et al.* Long-read-based human genomic structural variation detection with cuteSV. *Genome Biol* 2020;**21**:1–24.
31. Jiang T, Cao S, Liu Y, *et al.* Regenotyping structural variants through an accurate force-calling method. *bioRxiv* 2023; 2022.08.29.505534.
32. Heller D, Vingron M. SVIM: structural variant identification using mapped long reads. *Bioinformatics* 2019;**35**:2907–15.
33. Liu Y, Jiang T, Su J, *et al.* SKSV: ultrafast structural variation detection from circular consensus sequencing reads. *Bioinformatics* 2021;**37**:3647–9.
34. Cretu Stancu M, Van Roosmalen MJ, Renkens I, *et al.* Mapping and phasing of structural variation in patient genomes using nanopore sequencing. *Nat Commun* 2017;**8**:1326.
35. Ho SS, Urban AE, Mills RE. Structural variation in the sequencing era: comprehensive discovery and integration. *Nat Rev Genet* 2020;**21**:171–89.
36. Nagasaki M, Yasuda J, Katsuoka F, *et al.* Rare variant discovery by deep whole-genome sequencing of 1,070 Japanese individuals. *Nat Commun* 2015;**6**:8018.
37. Wall JD, Stawiski EW, Ratan A, *et al.* The GenomeAsia 100K project enables genetic discoveries across Asia. *Nature* 2019;**576**:106–11.
38. Wu D, Dou J, Chai X, *et al.* Large-scale whole-genome sequencing of three diverse Asian populations in Singapore. *Cell* 2019;**179**:736–749.e15.
39. The 100,000 Genomes Project Pilot Investigators. 100,000 genomes pilot on rare-disease diagnosis in health care—preliminary report. *New Engl J Med* 2021;**385**:1868–80.

40. Bolognini D, Sanders A, Korbel JO, *et al.* VISOR: a versatile haplotype-aware structural variant simulator for short- and long-read sequencing. *Bioinformatics* 2020;**36**:1267–9.
41. Li H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* 2018;**34**:3094–100.
42. Danecek P, Bonfield JK, Liddle J, *et al.* Twelve years of SAMtools and BCFtools. *GigaScience* 2021;**10**:giab008.
43. Zook JM, Hansen NF, Olson ND, *et al.* A robust benchmark for detection of germline large deletions and insertions. *Nat Biotechnol* 2020;**38**:1347–55.