Bioinformatics, 33(2), 2017, 192–201
doi: 10.1093/bioinformatics/btw594
Advance Access Publication Date: 25 September 2016
Original Paper

OXFORD

Sequence analysis

LAMSA: fast split read alignment with long approximate matches

Bo Liu†, Yan Gao† and Yadong Wang*

Center for Bioinformatics, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.
Associate Editor: John Hancock

Received on December 8, 2015; revised on July 20, 2016; accepted on September 8, 2016

Abstract

Motivation: Read length is continuously increasing with the development of novel high-throughput sequencing technologies, which has enormous potentials on cutting-edge genomic studies. However, longer reads could more frequently span the breakpoints of structural variants (SVs) than that of shorter reads. This may greatly influence read alignment, since most state-of-the-art aligners are designed for handling relatively small variants in a co-linear alignment framework. Meanwhile, long read alignment is still not as efficient as that of short reads, which could be also a bottleneck for the upcoming wide application.

Results: We propose long approximate matches-based split aligner (LAMSA), a novel split read alignment approach. It takes the advantage of the rareness of SVs to implement a specifically designed two-step strategy. That is, LAMSA initially splits the read into relatively long fragments and co-linearly align them to solve the small variations or sequencing errors, and mitigate the effect of repeats. The alignments of the fragments are then used for implementing a sparse dynamic programming-based split alignment approach to handle the large or non-co-linear variants. We benchmarked LAMSA with simulated and real datasets having various read lengths and sequencing error rates, the results demonstrate that it is substantially faster than the state-of-the-art long read aligners; meanwhile, it also has good ability to handle various categories of SVs.

Availability and Implementation: LAMSA is available at https://github.com/hitbc/LAMSA

Contact: Ydwang@hit.edu.cn

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

1 Introduction

Developing novel high-throughput sequencing (HTS) technologies, such as Illumina Moleculo (http://www.illumina.com/technology/next-generation-sequencing/long-read-sequencing-technology.html), PacBio Single Molecular Real-time (SMRT) Sequencing (Eid *et al.*, 2009) and Oxford Nanopore Technologies (Eisenstein, 2012; Schneider and Dekker, 2012; Ip *et al.*, 2015), have increased read length to over thousands of base-pairs (bps). These innovative technologies are promoting cutting-edge genomic studies (Chaisson and Tesler, 2012; Koren *et al.*, 2012; Huddleston *et al.*, 2014; Sudmant *et al.*, 2015), while they also influence various kinds of HTS data analysis. One of the most seriously affected analyses is read alignment, i.e. aligning HTS reads to reference genome to determine the likely positions of the reads.

Read alignment is fundamental to the variant calling of resequenced genomes (Li *et al.*, 2008; McKenna *et al.*, 2010; DePristo *et al.*, 2011; Jiang *et al.*, 2012), and it is one of the most compute-intensive steps in HTS data analysis (Langmead *et al.*, 2012). Due to its widely application, many efforts have been made (Fonseca *et al.*, 2012); however, most state-of-the-art aligners, such as BWA (Li and Durbin, 2009), Bowtie (Langmead *et al.*, 2009), Bowtie2 (Langmead *et al.*, 2012), GEM (Marco-Sola *et al.*, 2012), SOAP3 (Liu *et al.*, 2012), STAR (Dobin *et al.*, 2012), SeqAlto (Mu *et al.*,

© The Author 2016. Published by Oxford University Press. All rights reserved. For Permissions, please e-mail: journals.permissions@oup.com

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

2012), are designed for the reads which are several tens to several hundreds of bps long (i.e. reads produced by popular platforms such as Illumina HiSeq, Roche 454, Ion Torrent, etc.). These aligners mainly aim at implementing the co-linear alignments between reads and reference genome. They have demonstrated their efficiency as well as ability of handling the co-linear and relatively small events within short reads, such as SNPs, indels and sequencing errors. However, they may be not well-suited to reads over thousands of bp long, since longer reads more frequently span the breakpoints of non-co-linear structural variations (SVs), such as inversions, translocations and duplications (Feuk *et al.*, 2006), which cannot be effectively handled by co-linear approach. There are also methods designed for the alignment of long genomic sequences, such as BLAT (Kent, 2002), SSAHA2 (Ning *et al.*, 2001) and LAST (Kiełbasa *et al.*, 2011). However, they more emphasize on the alignment of even longer genomic fragments but not HTS reads. Such approaches are sensitive but could be lack of efficiency for coping with large volume of HTS data.

State-of-the-art long read aligners are BWA-SW (Li and Durbin, 2010), BWA-MEM (Li, 2013), BLASR (Chaisson and Tesler, 2012) and YAHA (Faust and Hall, 2012). BWA-SW, BWA-MEM and BLASR are on the basis of seed-and-extension approach, i.e. they match partial reads ('seeds') to reference genome to find candidate positions ('hits'), and align the reads with the local sequences surrounding the hits ('extension') to compose whole read alignment. Their major difference is the way of seeding. BWA-SW uses the approximate matches between the read and reference genome as hits to find candidate local regions for extension. The bottleneck of this approach is that it is quite expensive to traverse the suffix-trie of reference genome to query the approximate matches. Both of BWA-MEM and BLASR find the exact matches of short tokens as hits, and further cluster the hits to determine the candidate local regions. The bottleneck faced by this approach is how to handle repetitive genomic regions. For example, more than half of the human genome is comprised of repeats (De Koning *et al.*, 2011; Treangen and Salzberg, 2012). In this context, a short token may have many matches (hits), and it could be expensive to sort, prioritize and evaluate the numerous hits. Besides the details of implementation, the three aligners are still based on the co-linear model. In consideration of the potential breakpoints within reads, they focus on producing high-quality local alignments for the reads, other than end-to-end alignments.

YAHA is also a seed-and-extension-based approach; however, it is designed for implementing non-co-linear alignment. Briefly, it uses the hits of short seeds to build a non-co-linear skeleton of alignment to represent the potential structural variation events. It then composes the whole read alignment by filling the gaps of the skeleton with anchored local alignment. This approach is more favorable to handle the non-co-linear events. However, it is also not efficient, likely due to two issues. Firstly, the seeds used by YAHA could be very repetitive due to its short length (typically 8–15 bp); and secondly, the cost of building the skeleton is even higher due to the many hits of the seeds.

Herein, we propose long approximate matches-based split aligner (LAMSA), a novel split read alignment approach which is suited to various error sequencing rates with faster speed as well as good ability of handling non-co-linear events. LAMSA adopts a two-step split read alignment strategy, which separately handles the co-linear and non-co-linear events within the reads. In this strategy, LAMSA splits a read into relatively long fragments and utilizes the co-linear alignments of the fragments (i.e. approximate matches) for building one or more alignment skeletons. Previous study (Lim *et al.*, 2015) indicated that, it is more feasible to recover the true locations of the read by using the alignments of relatively long fragments instead of low-distance short seeds, especially when there are a number of mismatches and/or indels in the read. LAMSA utilizes the alignments of the fragments to build high quality of alignment skeletons. In the context of the skeletons, LAMSA classifies the unaligned parts of the read into several categories of non-co-linear events and handles each of them with a specific split alignment method according to its category.

Although other extant aligners (e.g. YAHA and STAR) have implemented various approaches to use the matches of partial reads as anchors to handle large or non-co-linear events within the reads, LAMSA's two-step strategy has its own advantages. Other than that of short seeds, there would be less false positive matches to affect the alignment as long fragments are less repetitive. Moreover, due to that relatively few fragments are used, their matches can be more comprehensively considered without loss of efficiency. This is important since it could be also expensive to comprehensively analyze the matches of repetitive short seeds with complicated approach (e.g. the short seed-based alignment skeleton used by YAHA). Other simplified strategies are also proposed to speed up the processing of short seeds (e.g. the seed-and-stich strategy of STAR). However, LAMSA uses a sparse dynamic programming (SDP) process to efficiently build a relatively comprehensive set of candidate alignment skeletons, to fully consider the potential co-linear and non-co-linear events described by the alignments of long fragments.

We benchmarked LAMSA by simulated reads with various lengths and sequencing error rates, as well as real long reads produced by Illumina Moleculo, PacBio SMRT and Oxford Nanopore platforms. The results demonstrated that LAMSA can substantially improve the speed of the long read alignment; meanwhile, it also has good ability to handle the breakpoints of the reads and produce sensitive and accurate alignments.

2 Methods

2.1 Overview

LAMSA initially extracts a series of seeding fragments from the read, which are longer than the seeds commonly used by state-of-the-art methods. With these fragments, LAMSA implements the alignment of the read in two steps. In the first step, considering that SVs do not occur as frequently as that of small co-linear variants such as SNPs and small indels (Mills *et al.*, 2011), LAMSA assumes all fragments as SV-free at first, and employs extant short read aligner to query the approximate matches of the fragments on reference genome. The matches of the fragments are then processed by a SDP-based algorithm to compose a series of skeletons which consist of various sets of anchored matches and gaps. Each of the skeletons corresponds to a specific part of the read. In the second step, LAMSA separately fills the gaps within the skeletons. For each of the gaps, LAMSA classifies it into one of four categories (see Section 2.3.1). And then, for each of the categories, LAMSA implements a specific split-alignment strategy to fill the corresponding gaps. The whole read alignment is accomplished by integrating the skeletons and the alignments of the gaps. A schematic illustration of the LAMSA method is in Supplementary Figure S1.

2.2 Querying long approximate matches

LAMSA extracts a series of substrings from the read as 'seeding fragments' to build the skeleton of alignment. More precisely, LAMSA extracts the fragments starting at every *FI* bp of the read, each of the fragments is *FL* bp long (*FI* and *FL* are user-defined parameters).

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

For each of the fragments, all its approximate matches on the reference genome are queried to compose a comprehensive analysis in later steps. Theoretically, the query is an end-to-end alignment of the fragment against the reference genome which can be implemented by any state-of-the-art short read aligner. In consideration of efficiency, GEM mapper (with its fast mode) (Marco-Sola *et al.*, 2012) is employed.

2.3 Building the skeleton of alignment

Each of the matches of the fragments can be denoted as a tuple, $M_i(M_i^R, M_i^G, M_i^{aln})$, $i = 1, \ldots, N_M$, where M_i^R and M_i^G are respectively the starting positions of the match on the read and the reference genome; M_i^{aln} is the detail of the alignment, e.g. its Compact Idiosyncratic Gapped Alignment Report (CIGAR) string, and N_M is the total number of the matches of the fragments. In addition, we use $|M_i^{aln}|$ to denote the length of the match (alignment) on the reference genome. The length is determined by taking account all the matches, insertions and deletions which are depicted by the alignment, M_i^{aln}. With these matches, LAMSA builds a direct acyclic graph (DAG) and performs a specifically designed SDP method to generate not only the optimal, but a set of possible alignment skeletons. Each of the skeletons consists of a series of co-linear events (such as insertions and deletions) and/or non-co-linear events (such as duplications and inversions), which are represented by a path of the DAG, and all the skeletons are organized by a spanning tree. Further, LAMSA prunes the spanning tree to prioritize the skeletons.

2.3.1 Generating skeletons of alignment with SDP

LAMSA builds the approximate matches-based DAG at first. The vertices of the graph consist of all the approximate matches, plus an auxiliary vertex, $M_{\text{start}}(-FL, *, *)$ representing the start of the alignment. In the DAG, each of the edges, $M_i \rightarrow M_j$, connecting a pair of solid vertices (i.e. two matches) meets one of the following four conditions (Fig. 1), which corresponds to a specific co-linear or non-co-linear event.

a. Match (co-linear): $|\Delta M^G - \Delta M^R| \leq \varepsilon$.
b. Duplication (non-co-linear): $-\tau < \Delta M^G - \Delta M^R \leq -\theta$.
c. Deletion (co-linear): $\varepsilon < \Delta M^G - \Delta M^R < \tau$.
d. Insertion (co-linear): $-\Delta M^R < \Delta M^G - \Delta M^R < -\varepsilon$.

Here, M_i and M_j are two vertices other than M_{start}, $\Delta M^G = M_j^G - (M_i^G + |M_i^{aln}|)$, $\Delta M^R = M_j^R - (M_i^R + FL)$, and ε, τ and θ (default: $FI \times 4\%$, 10 000 and 50) are three user-defined parameters categorizing the edges. Here, ε is used to account for the small indels within the approximate match, and τ and θ are used to model specific categories of SVs. A schematic illustration of the four categories of edges is in Figure 1.

Each of the categorized edges is further assigned a score according to its category. The scores are also user-defined parameters, and their default values are respectively match edges: +1 (if $\Delta M^R \leq 10FI$) or −1 (if $\Delta M^R > 10FI$), duplication edges: −3, deletion/insertion edges: −3. In addition, for each of the solid vertices, M_i, there is also an auxiliary edge $M_{\text{start}} \rightarrow M_i$ with 0 score, which represents the start of some local alignment of the read.

With the scored edges, LAMSA performs a SDP (Supplementary Fig. S2) on the matches by the following recursion equation to generate the alignment skeletons:

$$P_V(M_j) = \max\{P_V(M_i) + P_E(M_i \rightarrow M_j)\}, M_i \rightarrow M_j \in E,$$

where $P_V(M_i)$ is the score assigned to M_i and $P_E(M_i \rightarrow M_j)$ is the score of the edge $M_i \rightarrow M_j$. With this equation, each vertex can find a precursor maximizing its score, and each path starting from M_{start} and ending at a solid vertex without successor formulates a skeleton.

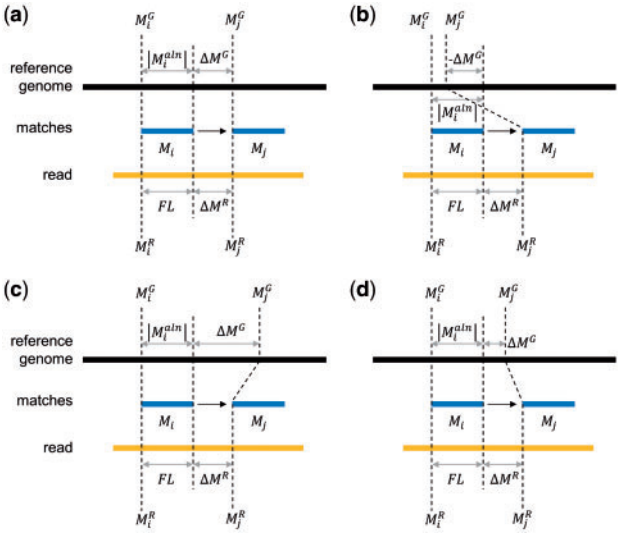

Fig. 1. A schematic illustration of the four categories of edges. (**a**) Match, (**b**) duplication, (**c**) deletion and (**d**) insertion. In the figure, the black, yellow and blue bars respectively indicate the reference genome, the read and the anchored approximate matches. The positions of the matches on the reference genome and the read are marked by dashed lines, which also illustrate the classification conditions (Color version of this figure is available at *Bioinformatics* online.)

It is also worth noting that SDP process does not explicitly consider the non-co-linear inversions, but models them implicitly. For an inversion, there would be two approximate matches flanking it, and their distances on the read and reference genome should be similar, as inversion is a kind of balanced SV (Supplementary Fig. S3). In this situation, the two approximate matches would form a 'match' edge. Furthermore, in most cases, as *FI* and *FL* parameters are much smaller than the size of the inversion, the corresponding read part can fully cover one or more seeding fragments. These fragment(s) will be matched to the strand of the reference genome inverse to that of the flanking approximate matches, and constitute a standalone skeleton. Thus, in later steps, with the help of the skeleton, the read part flanking and involved in the inversion event would be correctly aligned to reference genome.

2.3.2 Prioritizing candidate skeletons

As each of the vertices of the DAG has only one precursor after the SDP, LAMSA organizes all the skeletons with the spanning tree of the DAG. The head node of the tree is M_{start}, and for each of the approximate matches (solid vertices), its parent node is its precursor determined by the SDP (Supplementary Fig. S4a). Thus, each path from the head node to a leaf node represents a skeleton. Moreover, each path of the tree can be scored by summing up the scores of the edges involved in the path.

LAMSA prioritizes the skeletons by pruning the spanning tree. Before the pruning, each of the non-branched paths of the tree is collapsed as a single node (Supplementary Fig. S4b). The pruning is iteratively performed. In each iteration, a leaf node with highest score is selected, and if it does not directly connect to the head node, LAMSA connects all its sibling nodes to the head node, and merges it to its parent node. Each of the pruned sibling nodes is rescored by the edges involved in the affiliated path of the node. This operation is iteratively performed until all the nodes other than the head node become leaf nodes (Supplementary Fig. S4c, d). After the pruning, each of the leaf nodes depicts a specific skeleton. Moreover, it is

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

worth noting that, the global best result of the SDP, i.e. the path of the DAG with the highest score will be kept as a unpruned skeleton, since in each iteration of the pruning, the leaf node corresponding to this skeleton has a higher score than its siblings.

Further, LAMSA uses a greedy strategy to cluster the recorded skeletons. That is, LAMSA sorts all the skeletons by their ending positions on the read, and initially selects the skeleton whose ending position is closest to the right end of the read as the 'seeding skeleton' to generate a new cluster. Given a cluster, LAMSA investigates each of the skeletons not in the cluster if over S_F% (default: 70%) of the fragments covered by the skeleton are also covered by the cluster. If so, LAMSA adds the skeleton into the cluster, and updates the fragment set of the cluster. If there is no skeleton can be added into the cluster, LAMSA set the cluster aside, and picks out the remaining skeleton which is closest to the right end of the read as the seeding skeleton to build a new cluster and continue the processing, until no skeleton remains.

LAMSA prioritizes the skeletons by their scores. For each of the clusters, only its first N_s skeletons are recorded and other skeletons are set aside. The parameter N_s is empirically selected as 10 in practice. After the clustering, each cluster covers a specific read part, and the skeletons of the cluster would be further processed to generate the primary and alternative alignments for the read part.

2.4 Filling the gap of skeleton

LAMSA then fills the gaps within the skeletons to solve the breakpoints of SVs and generate valid alignments for the whole read. Each of the gaps is filled by one of the following strategies according to its category.

2.4.1 Filling a match gap

For a 'match' edge, if the two anchoring matches correspond to two neighboring seeding fragments of the read, LAMSA directly performs an end-to-end alignment between the gapped part of the read and the anchored reference sequence to fill the gap (Supplementary Fig. S5a).

If the two anchoring matches are not corresponding to two neighboring seeding fragments on the read, LAMSA separately extend the upstream and downstream matches with Smith–Waterman (SW) algorithm at first (Supplementary Fig. S5b). If the extended alignment from either the upstream or the downstream match can cover over half of the gapped part, LAMSA end-to-end realigns the gapped part against the anchored local reference sequence. Otherwise, LAMSA separately keeps the upstream and downstream extended alignments as a result, and records the read part still unaligned for further processing.

2.4.2 Filling a duplication gap

For a 'duplication' edge, LAMSA separately handles the upstream and downstream vertices. That is, LAMSA extracts the local reference sequences surrounding the matches of the upstream and downstream fragments at first (Supplementary Fig. S6). For each of the two fragments, LAMSA extends its match by aligning the gapped part of the read against the corresponding local reference sequence, i.e. the upstream match is extended downstream and the downstream match is extended upstream. With this strategy, the two vertices of the edge are treated as two copies of the same local reference sequence, and each of them is handled by a specific local alignment.

Some duplication events may involve more than two copies of the same local reference sequence. In this situation, there will be not only one, but a series of duplication edges. LAMSA will separately process each of those edges with this strategy.

2.4.3 Filling a deletion gap

For a 'deletion' edge, LAMSA builds a hash table on-the-fly for the all the l-mers of the junction sequence of reference genome, and matches all the l-mers of the gapped read part to the junction (Supplementary Fig. S7). A SDP based on the l-mer matches is implemented to build a local skeleton of the alignment which can maximize the total number of matched read bases. This l-mer-based skeleton is beneficial for handling multiple breakpoints within the junction. For example, if there are more than one deletions within the junction, the skeleton can effectively recognize them.

Given the l-mer-based skeleton, LAMSA categorizes each of the gaps with the conditions similar to the four conditions applied to approximate matches-based skeletons. For a gap recognized as match, deletion or insertion, the gapped read part is end-to-end aligned to local reference sequence anchored by the corresponding l-mer matches; while for a duplication gap, LAMSA fills it with the strategy similar to that of the duplication gaps within approximate matches-based skeletons.

2.4.4 Filling an insertion gap

For an 'insertion' edge, the operation is opposite to that of 'deletion' edges. That is, LAMSA treats the gapped part of the read as the junction sequence to build the l-mer-based skeleton, and the anchored reference sequence is aligned to the gapped part of the read.

2.4.5 Extending the boundaries of the skeletons

All the operations mentioned above fill the inner gaps of the skeleton, which are anchored by two approximate matches. In addition, LAMSA also extends the outer boundaries, i.e. the matches at the ends of the skeletons, which can be also seen as gaps with only one anchor. For each of the boundaries, LAMSA assumes that the neighboring genomic region is SV-free, and directly extends the match by SW algorithm.

2.5 Additional processing

LAMSA collects the generated split alignments, i.e. the approximate matches of the seeding fragments and the alignments of the gapped parts of the read. The alignments adjacent on both the read and the reference genome are chained to build more consecutive alignments. When chaining the alignment of a gapped read part with a neighboring approximate match, LAMSA checks if there is one or more indels around the boundary between the gapped read part and the seeding fragment. If this is the case, LAMSA would realign the read part around the boundary to reduce some false positives (Supplementary Fig. S8).

For a very small proportion of reads, there may be still some unaligned parts left after all the processing mentioned above. These unaligned parts are further handled with two additional steps. Firstly, for each of these unaligned parts, LAMSA investigates if it can be covered by the skeletons initially filtered out. If so, LAMSA fills the gaps within the corresponding skeleton(s) by the same methods mentioned above to generate the alignment(s) of the read part. Secondly, for each of the remaining unaligned parts, if it is short (no longer than 300 bp), LAMSA extracts all the k-mers of the part (default: $k = 19$) and queries all the exact matches of the k-mers with a FM-index (Ferragina and Manzini, 2000; Li and Durbin, 2009) of the reference genome. These matches are clustered by their

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

genomic positions, and LAMSA performs local alignments around the clusters to align the unaligned read parts.

3 Results

We benchmarked LAMSA with a series of simulated and real datasets. The speed, sensitivity and accuracy of the alignment were assessed and compared with four state-of-the-art long read aligners, BWA-SW, BWA-MEM, YAHA and BLASR, and a short read aligner, STAR, which is also applicable to long reads. All the benchmarks were implemented on a server with an Intel Xeon E4820 CPU at 2.00 GHz and 1 Terabytes RAM, running Linux Ubuntu 14.04. The command lines and versions of the aligners are available in Supplementary Notes.

3.1 Simulation study

In the simulation study, we generated a donor genome with 4002 SVs by RSVSim (Bartenhagen and Dugas, 2013). The ratio and the size of the integrated SVs are configured by referring to the DGV database (MacDonald *et al.*, 2014). Using the simulated donor genome, we generated 21 simulated datasets. Fifteen of them are simulated by Wgsim (Li et al., 2009) with low-sequencing error rates (1, 2 and 4%) and five kinds of read lengths (from 5000 to 100 000 bp), which mimics high quality sequences, such as Illumina Moleculo reads, error corrected PacBio SMRT reads or assembled contigs. Six of them were simulated by PBSim (version 1.0.3) (Ono et al., 2013) with high-sequencing error rates (10, 15, 20 and 30%) and three kinds of mean read lengths (2000, 3000 and 10 000 bp), which mimics the noisy long reads produced by PacBio SMRT and Oxford Nanopore platforms. More detailed information is in Supplementary Notes and Supplementary Tables S1 and S2.

3.1.1 Assessment on the speed

The speeds of the aligners indicate that, LAMSA overall has a substantial improvement on the speed (Fig. 2, Supplementary Tables S3 and S4, Supplementary Figs. S9 and S10). On the datasets with low-error rates (i.e. 1, 2 and 4%), LAMSA is overall about 7–12-folds faster than BWA-MEM and BWA-SW, and many 10-folds faster than YAHA, STAR and BLASR. Moreover, as YAHA does not support reads longer than 32 000 bp, we did not run it on the 50 000 bp and 100 000 bp datasets; and STAR showed a very low sensitivity (see later) on the 10 000 bp datasets, indicating that it may be not suited to very long reads. Meanwhile, it unaligned all the reads longer than 10 000 bp, thus only its results on the 5000 bp and 10 000 bp reads are shown. On the high-error rate datasets, LAMSA also outperforms BLASR and BWA-MEM by several folds (other aligners are not applicable to such high-error rates).

LAMSA gains the speed by that the long fragments are less repetitive than commonly-used short seeds, and the employed short read aligner can efficiently query the approximate matches. Thus, the problem can be quickly reduced, i.e. in most cases, only a few skeletons are left after the first step. In the second step, LAMSA can further handle the skeletons with efficiency, as the major cost of this step only originates from the alignments between a batch of gapped read parts and anchored local reference sequences, which are not expensive.

We also investigated the memory footprint of LAMSA. As both of the GEM mapper and the main part of the LAMSA method use FM-index to index the reference genome, it does not need large RAM space. On all the simulated datasets, the peak memory footprint is about 6.5 Gigabytes, which can be easily met by a modern PC.

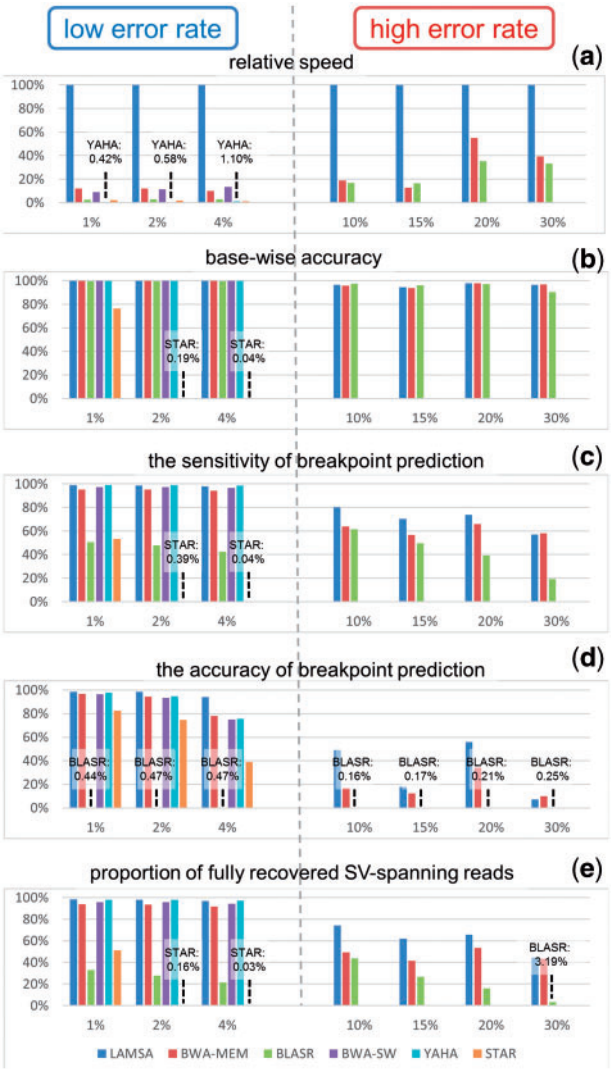

Fig. 2. Benchmarking on the low- and high-error-rate 10 000 bp datasets. (**a**) The relative speed of the aligners, which is defined as T_{LAMSA}/T_{aln}, where T_{LAMSA} and T_{aln} are respectively the alignment times of LAMSA and an compared aligner. (**b**) The base-wise accuracy. The base-wise accuracy is defined as the proportion of bases which are mapped to a position within 5 bp of its ground truth position. (**c**) The proportion of recovered ground truth breakpoints, which is calculated by N^R_{BP}/N^T_{BP}, where N^R_{BP} and N^T_{BP} are respectively the numbers of the ground truth breakpoints recovered by the alignments of the reads and all the ground truth breakpoints. (**d**) The proportion of correctly predicted breakpoints, which is calculated by N^C_{BP}/N^P_{BP}, where N^C_{BP} and N^T_{BP} are respectively the numbers of correctly predicted breakpoints and all the predicted breakpoints. (**e**) The proportion of the SV-spanning reads which are fully recovered by the aligner. Here, a SV-spanning read being 'fully recovered' indicates that all its breakpoints are recovered by the alignments. On some measures, some aligners have very low values, and they are shown by the precise numbers with dashed lines. More detailed results are in Supplementary Tables S3–S6 and Supplementary Figures S9 and S10 (Color version of this figure is available at *Bioinformatics* online.)

3.1.2 Assessment on the base-wise accuracy

We assessed the base-wise accuracy to investigate the overall quality of the alignments. We assess the accuracy by $Acc^{base} = N^{mat}_{aln}/N^{mat}_{truth}$, where N^{mat}_{truth} is the number of matched bases of the ground truth alignments of the reads; and N^{mat}_{aln} is the number of matched bases

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

which are aligned within T^{mat}_{diff} bp of the corresponding ground truth positions. We set $T^{mat}_{diff} = 5$ for all the datasets at first, while stricter thresholds ($T^{mat}_{diff} = 1$ and $T^{mat}_{diff} = 3$) were also used for the low-error rate datasets.

On the low-error rate datasets (Supplementary Table S3), LAMSA has overall the highest number of correctly aligned bases, while other aligners except STAR also achieved similar results. The results of STAR are poor, mainly due to that it has many unaligned bases. As STAR is mainly designed for short reads, the results are likely due to its inherent design and implementation that may be not suited to very long reads.

On the high-error rate datasets (Supplementary Table S4), the base-wise accuracies of LAMSA, BWA-MEM and BLASR are quite close, only except for the 30% dataset, where BLASR's accuracy dropped a little (i.e. 90.24%), but the accuracies of LAMSA and BWA-MEM do not change much (i.e. 96.47 and 96.82%, respectively).

The base-wise measures indicate that, overall, LAMSA can produce high-quality alignment for both of low- and high- error rate reads. However, it is also worth noting that, the base-wise accuracies of the aligners partially depend on the parameters used for the alignment at local genomic region. Moreover, local realignment is also usually performed in downstream SV analysis to further improve the accuracy of alignment. Under this circumstance, the difference on the accuracies of LAMSA and other aligners as well as the effect of this difference could be even smaller. However, considering relatively high accuracy achieved by LAMSA, it could be also beneficial to use the alignment of LAMSA as an input in downstream steps of SV analysis, such as realignment.

3.1.3 Assessment on the ability of handling SV

For assessing the ability of handling SVs, we compared the breakpoints of the reads predicted by the alignments with the ground truth. The predicted breakpoints are described by the $\geq T^{SV}_{indel}$ bp indels, and $\geq T^{SV}_{clip}$ bp clippings within the primary alignments of the reads. This is except for BLASR that the breakpoints described by all the alignments of BLASR are considered. This is due to that, for a given read, BLASR chooses only one alignment from all the alignments of all the split read parts as the primary alignment. But other aligners, such as LAMSA and BWA-MEM, differentiate primary and secondary alignments for each of the split read parts. Thus, we take account all the alignments of all the split read parts to avoid underestimation on the sensitivity of BLASR.

Each of the predicted breakpoints can be denoted as a tuple, $BP_i(\mathrm{Pos_{Read}}, \mathrm{Pos_{Ref}}, \mathrm{D_{BP}}),\ \mathrm{i} = 1, \ldots, N^P_{BP}$, where $\mathrm{Pos_{Read}}$ and $\mathrm{Pos_{Ref}}$ are respectively the coordinates of the breakpoints on the read and reference genome, D_{BP} is the mark indicating the corresponding SV event happens upstream or downstream $\mathrm{Pos_{Ref}}$, and N^P_{BP} is the total number of predicted breakpoints within the corresponding read. Similarly, each of the breakpoints described by the ground truth can be denoted as a tuple, $BP^T_i(\mathrm{Pos^T_{Read}},\ \mathrm{Pos^T_{Ref}}, D^T_{BP}),\ \mathrm{i} = 1, \ldots, N^T_{BP}$, where $\mathrm{Pos^T_{Read}}$, $\mathrm{Pos^T_{Ref}}$ and D^T_{BP} are the elements corresponding to that of a predicted breakpoint but given by the ground truth, while N^T_{BP} is the total number of ground truth breakpoints within the read.

We assessed the number of predicted breakpoints, the number of ground truth breakpoints being recovered, and the number of correctly predicted breakpoints of the aligners. For a certain read, a ground truth breakpoint, $BP^T_i(\mathrm{Pos^T_{Read}},\ \mathrm{Pos^T_{Ref}}, D^T_{BP})$ is considered as being recovered, only if there is at least one predicted breakpoint, $BP_i(\mathrm{Pos_{Read}}, \mathrm{Pos_{Ref}}, D_{BP})$, meeting the condition: $|\mathrm{Pos_{Read}} - \mathrm{Pos^T_{Read}}| < \mathrm{T^{SV}_{diff}}$, $|\mathrm{Pos_{Ref}} - \mathrm{Pos^T_{Ref}}| < \mathrm{T^{SV}_{diff}}$, and $D_{BP} = D^T_{BP}$; and a predicted breakpoint, $BP_i(\mathrm{Pos_{Read}}, \mathrm{Pos_{Ref}})$ is considered as correct, only if there is at least one ground truth breakpoint, $BP^T_i(\mathrm{Pos^T_{Read}},\ \mathrm{Pos^T_{Ref}}, D^T_{BP})$, meeting the above condition.

Considering that error prone reads are more likely to be clipped and more difficult to correctly align, we used different thresholds for the datasets. For low-error rate datasets, the thresholds are stricter, i.e. $\mathrm{T^{SV}_{indel}} = 50$, $\mathrm{T^{SV}_{clip}} = 10$, $\mathrm{T^{SV}_{diff}}$ =1 and 10; while for high-error rate datasets, $\mathrm{T^{SV}_{indel}} = 50$, $\mathrm{T^{SV}_{clip}} = 50$, $\mathrm{T^{SV}_{diff}}$ =1, 50 and 100 are used.

On the low-error rate datasets, the results demonstrate that LAMSA and YAHA recovered similar numbers of breakpoints (Fig. 2c and Supplementary Table S5), slightly more than those of BWA-MEM and BWA-SW, suggesting that they have better sensitivity for handling SVs. Moreover, YAHA has lower accuracy than that of LAMSA, especially for 4% error rate datasets (Fig. 2d and Supplementary Table S5). This is mainly due to that it aligns some of the SV-free parts of the reads by split alignments, thus more false positive breakpoints are produced.

In addition, we also separately assessed the numbers of recovered breakpoints by various categories of SVs (i.e. insertion, duplication, deletion and inversion) (Supplementary Table S5). The results indicate that all the aligners recovered most of the breakpoints of the insertions/duplications and deletions; however, BWA-MEM and BWA-SW recovered less breakpoints of the inversion events than that of LAMSA, especially on longer datasets. This is likely due to their design, i.e. co-linear local alignment, which could be hard to handle non-co-linear events. An example of the difference between the aligners on the handling of inversions is in Supplementary Figure S11.

We further assessed the ability of the aligners to fully recover the SV-spanning reads. Here, a SV-spanning read being 'fully recovered' indicates that all its breakpoints can be recovered by the alignments. This measure is important to haplotyping, a major application of long reads. The results (Fig. 2e and Supplementary Table S5) suggest that LAMSA and YAHA fully recovered similar numbers of SV-spanning reads, outperforming BWA-MEM and BWA-SW by a couple of percentages.

On the high-error rate datasets, overall LAMSA outperforms BWA-MEM and BLASR on sensitivity. On the 10, 15 and 20% datasets, LAMSA also achieves highest accuracy, and its accuracy on the 30% dataset is also close to that of the best one (BWA-MEM). Moreover, LAMSA fully recovers highest number of SV-spanning reads. These results suggest the good ability of LAMSA to handle the SVs within noisy long reads. Meanwhile, it is worth noting that, on both of low- and high- error rate datasets, BLASR reported many breakpoints (Supplementary Tables S5 and S6), which makes low accuracies.

Moreover, it is also observed that all the three aligners have two problems to handle the high error rate SV-spanning reads as follows.

Firstly, some of the breakpoints are hard to recover. This is mainly due to that, with the serious noise, some read parts are unaligned due to lack of anchors. This usually happens when the read parts are short and flanked with two breakpoints. In this situation, the anchors of other read parts also cannot help much, because the extensions from those anchors cannot span the breakpoints to align the no-anchor parts.

Secondly, there are much more false positive predicted breakpoints. This is mainly due to that with more error-tolerant parameters, some of the read parts could be aligned to many genomic positions by equal scores (although all these scores are low). These read parts usually have serious sequencing errors, and it is non-trivial to confidently align them.

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

It is also worth noting that, with stricter threshold, i.e. $T_{diff}^{SV}=1$, all the aligners achieved lower sensitivities and accuracies. This is largely caused by the implementations of local alignment of the read, e.g. the relatively simple scoring systems (such as user-defined local alignment scoring parameters), could not be able to well handle all sorts of SV events, thus some of the predicted breakpoints could be a little distant to the corresponding ground truth breakpoints. In this situation, local realignment may be necessary to further improve the quality of the alignment around the breakpoints to better support downstream SV analysis.

3.2 Real study

We used three datasets (Supplementary Table S1) respectively from Illumina Moleculo, PacBio SMRT and Oxford Nanopore platforms to benchmark the aligners on real datasets. The Illumina Moleculo dataset is a 40X coverage sequencing of the CEU HapMap individual NA12878; the PacBio SMRT dataset is a 10X coverage sequencing of the CHM1 cell line; the Oxford Nanopore dataset is a target sequencing of the CYP2D6, HLA-A, HLA-B genes of NA12878. The availability of the datasets is in Supplementary Notes, and the read length distributions are in Supplementary Figure S12 (assessed by FastQC software: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/).

3.2.1 Assessment on the speed

On all the three datasets, LAMSA has the fastest speed (Table 1). On the low-error rate Illumina Moleculo dataset, LAMSA is at least 4.6 times faster than the best runner-up (i.e. BWA-MEM), and about 8.6–123.3 times faster than the other aligners. On the noisy PacBio SMRT and Oxford Nanopore reads, LAMSA is about 2–4 times faster.

3.2.2 Assessment on the base-wise sensitivity and accuracy

Due to the lack of ground truth, we only assessed the base-wise sensitivity of the aligners on the Illumina Moleculo and PacBio SMRT datasets by the number of aligned bases. The results (Table 1) suggest that LAMSA and BWA-MEM aligned almost all the bases of the Illumina Moleculo reads, outperforming other aligners. And LAMSA aligned more bases of the PacBio SMRT reads than BWA-MEM and BLASR.

Table 1. The speed and base-wise sensitivity/accuracy on real datasets

Aligner	Aligned bases		Running time (seconds)
	#	%	
Illumina Moleculo dataset, 22 721 139 reads/91 476 572 938 bases			
LAMSA	90 838 105 363	99.30	106362.5
BWA-MEM	90 959 839 527	99.44	490067.6
BWA-SW	83 057 280 108	90.80	922687.3
YAHA	86 519 067 187	94.58	13120906.9
STAR	87 336 478 637	95.47	1214121.7
BLASR	90 173 742 575	98.58	4628608.8
PacBio SMRT dataset, 4 395 201 reads/32 525 651 939 bases			
LAMSA	31 156 436 703	95.79	457338.4
BWA-MEM	30 338 584 767	93.28	1050769.6
BLASR	29 475 310 169	90.62	1900594.2
Oxford Nanopore dataset, 7540 reads/26 279 177 bases			
LAMSA-BLASR	18 685 475	71.10	327.5
LAMSA-BWA	16 558 323	63.01	313.3
BWA-MEM	17 615 965	67.03	796.6
BLASR	13 643 127	51.92	1259.4

For the Oxford Nanopore dataset, the 'aligned bases' indicates the bases which are aligned to the regions of CYP2D6, HLA-A, HLA-B genes. 'LAMSA-BLASR' and 'LAMSA-BWA' respectively indicates LAMSA running with the SW parameters of BLASR and BWA-MEM.

As the Oxford Nanopore dataset is a target sequencing dataset, the accuracy of the alignment can be partially assessed by the bases aligned to the regions of CYP2D6, HLA-A, HLA-B genes. When aligning the nanopore reads, we configured LAMSA with two sets of SW parameters which are respectively the same to that of BWA-MEM (i.e. match = 1, mismatch = −1, gap-open = −1 and gap-extension = −1) and BLASR (i.e. match = 5, mismatch = −6, gap-open = 0 and gap-extension = −5). The result (Table 1) indicates that with the BLASR-like parameters, LAMSA aligned more bases to the correct regions than the other two aligners.

3.2.3 Assessment on the ability of handling SV

On the high coverage Illumina Moleculo and PacBio SMRT datasets, we assessed the aligners' ability of SV-handling. Since it is lack of the ground truth for the alignment, for the Illumina Moleculo dataset, we compared the predicted breakpoints with the breakpoints reported in the 1000 Genomes Project Phase 3 release (Sudmant *et al.*, 2015) (short as '1KG' breakpoints, totally 6576 derived from 3260 SVs); and for the PacBio SMRT dataset, we compared the predicted breakpoints with the breakpoints reported by a previous study on CHM1 cell line (Chaisson *et al.*, 2015) (short as 'CHM1' breakpoints, totally 37 150 derived from 18 542 >50 bp SVs). The availability of the 1KG and CHM1 breakpoints is in Supplementary Notes.

The numbers of recovered 1KG/CHM1 breakpoints were assessed at first. More precisely, a 1KG/CHM1 breakpoint is termed as $BP^G(\mathrm{Pos}_{\mathrm{Ref}}^G, D_{BP}^G)$, since it only has the genomic position; and it is considered as being recovered only if there is at least one predicated breakpoint $BP(\mathrm{Pos}_{\mathrm{Read}}, \mathrm{Pos}_{\mathrm{Ref}}, D_{BP})$ within all the primary alignments of the reads, which meets the following condition: $|\mathrm{Pos}_{\mathrm{Ref}} - \mathrm{Pos}_{\mathrm{Ref}}^G| < T_{dis}$ and $D_{BP} = D_{BP}^G$, where T_{dis} is the threshold of the distance.

On the Illumina Moleculo dataset, we used a series of T_{dis} thresholds (i.e. 0, 10, 50 and 100 bp) to evaluate the alignments (Table 2). We found that all the aligners recovered similar number of breakpoints, while LAMSA recovered most breakpoints at 0 distance ($T_{dis}=0$). For more relaxed but still strict thresholds, such as $T_{dis}=10$, the numbers of the breakpoints recovered by LAMSA are also highest. This indicates that LAMSA can recover the breakpoints sensitively and precisely.

Meanwhile, we also assessed the numbers of predicted breakpoints which can be validated by the 1KG/CHM1 breakpoints, to investigate the accuracy of the aligners on handling SVs. Here, a predicted breakpoint, $BP(\mathrm{Pos}_{\mathrm{Read}}, \mathrm{Pos}_{\mathrm{Ref}}, D_{BP})$, is considered as validated, only if there is at least one 1KG/CHM1 breakpoint $BP^G(\mathrm{Pos}_{\mathrm{Ref}}^G, D_{BP}^G)$ which also meets the condition mentioned above.

On the Illumina Moleculo datasets, we observed that at very strict thresholds, e.g. $T_{\mathrm{dis}}=0$, the aligners have various numbers of validated breakpoints. This is likely due to the various scoring systems of the aligners. Considering that SV calling usually not only depends on read alignment, but also many downstream analysis steps, such as realignment and local assembly, it is also important to pay attention to the number of validated breakpoints at more relaxed but still strict threshold, such as $T_{\mathrm{dis}} \geq 10$, because the downstream

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

steps usually consider all the alignments around the breakpoints. At $T_{\text{dis}} = 10$, LAMSA, BWA-SW and STAR have similar numbers of validated breakpoints, suggesting that they have similar ability of SV handling. For YAHA, it has even more breakpoints being validated. However, it has about 4-folds more predicted breakpoints than that of LAMSA and BWA-MEM, indicating that the specificity of the alignment of YAHA could be lower.

We also simulated 10 sets of random breakpoints by RSVSim to test if the validated breakpoints predicted by the aligners are by chance. Each set has the same number of (i.e. 6576) breakpoints to that of the 1KG breakpoints. For each set, we assessed the numbers of predicted breakpoints matched by the random breakpoints in various T_{dis} thresholds, and calculated the average numbers across various sets. It can be observed from the results (Supplementary Table S7) that, even if the threshold is very large ($T_{\text{dis}} = 1000$), only a very small proportion (e.g. <0.001% for LAMSA) of predicted breakpoints can be accidentally matched by the random breakpoints. This suggests that most of validated breakpoints are not obtained by chance, but reasonable alignments.

To assess the overall ability of the SV handling, the *F*-scores of the aligners at a moderate strict threshold ($T_{\text{dis}} = 10$) were further calculated by $F_{BP} = 2\text{Sen}^{BP} \cdot \text{Acc}^{BP} / (\text{Acc}^{BP} + \text{Sen}^{BP})$, $\text{Sen}^{BP} = N_{re}^{BP} / N_{G}^{BP}$, $\text{Acc}^{BP} = N_{\text{val}}^{BP} / N_{\text{pre}}^{BP}$, where N_{re}^{BP}, N_{G}^{BP}, N_{val}^{BP} and N_{pre}^{BP} are respectively the numbers of the recovered, ground truth (1KG), validated and predicted breakpoints. The results (Table 3) indicate that STAR achieved a relatively higher *F*-score. Meanwhile, the *F*-scores of LAMSA and BWA-MEM are similar, which is higher than that of YAHA. Moreover, it is also very worth noting that, for all the aligners, the *F*-scores are quite low. This is mainly due to that there are many potentially false positive breakpoints, i.e. the predicted breakpoints which cannot be validated.

A major cause for this issue is the repetitive reads. It is observed that, a small proportion (0.36% for LAMSA, Supplementary Tables S8 and S9) of reads have many (> 10) predicted breakpoints. The total number of these breakpoints is large and most of them are false positive. Most of such reads have some parts which can be split aligned to a number of positions, and this is hard for the aligners to make confident choice. Aligners like LAMSA, BWA-MEM and BWA-SW aligned these repetitive parts to many positions with similar scores. This lowered down the Acc^{BP} values as well as the *F*-scores. STAR usually aligned such read parts to one or a few positions, while it also unaligned some of them. With this strategy, STAR achieved slightly higher Acc^{BP}, but lower base-wise sensitivity. The strategy of YAHA is similar to that of LAMSA and BWA-MEM, but it also 'over-split aligned' some of the reads which are consecutively aligned by other aligners, i.e. YAHA splits them into many small pieces, and maps them to a number of positions (an example is in Supplementary Fig. S13).

In the reads with less (≤ 10) predicted breakpoints, there are also breakpoints cannot be validated (Supplementary Tables S8 and S9). We investigated the details of the alignments, and found that this could be due to two issues. Firstly, for some of the reads, the aligners split them into parts and aligned each of the parts to a unique position with small edit distance. These alignments could be also reasonable, especially considering that multiple aligners independently generated similar alignments. These breakpoints could be of unknown SVs, as the ground truth SV list could be still not complete. Secondly, for some of the reads, the aligners also split aligned them, but some parts were aligned with very low scores. These poorly aligned parts could be of very complicated SVs that cannot be well-handled by the aligners, or the sequences which are not suitable to align to the reference, such as novel insertions. Moreover, it could be also possible that some of these read parts have exceptional low sequencing quality, although the overall sequencing quality of the whole dataset is high.

On the PacBio SMRT dataset, we used the same T_{dis} thresholds for the evaluation (Table 2). However, it is observed that at very strict

Table 2. The numbers of recovered breakpoints

Aligner	# of recovered breakpoints	# of the recovered breakpoints within various distance thresholds (bp)			
		0	≤10	≤50	≤100
Illumina Moleculo dataset (6576 1KG breakpoints in total)					
LAMSA	5687	3684	4674	5612	5687
BWA-MEM	5560	2811	3882	5443	5560
BWA-SW	5511	2458	3700	5370	5511
YAHA	5579	2463	3722	5426	5579
STAR	5547	3181	4296	5448	5547
BLASR	5340	2138	4411	5467	5603
PacBio SMRT dataset (37 150 CHM1 breakpoints in total)					
LAMSA	32 525	2977	18 903	29 889	33 357
BWA-MEM	21 949	1682	11 080	18 379	21 949
BLASR	33 700	2733	21 315	31 238	33 700

Table 3. The numbers of predicted and validated breakpoints

Aligner	# of predicted breakpoints	# of the validated breakpoints within various distance thresholds (bp)				*F*-score at T_{dis}=10
		0	≤10	≤50	≤100	
Illumina Moleculo dataset (6576 1KG breakpoints in total)						
LAMSA	5 154 402	39 191	75 091	100 982	105 050	0.0286
BWA-MEM	5 114 765	49 860	70 358	97 368	106 140	0.0269
BWA-SW	3 920 507	38 155	60 307	86 717	90 650	0.0299
YAHA	22 471 827	54 499	81 029	114 409	126 854	0.0072
STAR	1 770 061	56 412	78 393	95 438	97 007	0.0830
BLASR	31 524 631	20 445	63 491	91 443	97 855	0.0040
PacBio SMRT dataset (37 150 CHM1 breakpoints in total)						
LAMSA	25 502 239	4387	69 489	203 601	338 603	0.0054
BWA-MEM	5 171 459	2664	40 131	89 816	139 240	0.0153
BLASR	33 980 974	4474	82 280	27 4720	43 0665	0.0048

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

thresholds, such as $T_{dis} = 0$, all the three aligners recovered low numbers of CHM1 breakpoints. This is likely due to that the serious sequencing errors affect the alignment. For example, with the scoring systems of the aligners, the read parts around the breakpoints are much easier to be largely clipped, so that the predicted breakpoints are more distant to the ground truth breakpoints. However, on more relaxed thresholds, the numbers of recovered CHM1 breakpoints obviously increase, indicating that only the bases very close to the breakpoints are hard for the aligners to handle, while most bases can still be appropriately aligned. Overall, LAMSA and BLASR recovered comparable numbers of CHM1 breakpoints, suggesting that their sensitivities are close, but the sensitivity of BWA-MEM is lower, i.e. it recovered fewer CHM1 breakpoints.

LAMSA and BLASR have more validated as well as predicted breakpoints than that of BWA-MEM. This is mainly due to that some of the read parts are repetitive and highly error-prone. In this situation, the read parts can be aligned to many positions with equally high scores, so that LAMSA and BLASR have more predicted breakpoints. But BWA-MEM clipped (unaligned) a portion of the error-prone read parts. This indicates a different design to LAMSA in the consideration of the balance between sensitivity and specificity, as LAMSA usually tries to align the bases as many as possible to produce sensitive alignments.

4 Discussion

With the development of HTS technologies, it is expectable that both the read length and the throughput will continuously increase in the future. In the consideration of its enormous potentials, it is important to develop novel long read alignment tool with fast speed as well as good ability to handle SVs, as read alignment is the most compute-intensive step in resequencing studies, and long reads will more frequently span the breakpoints of SVs.

We developed LAMSA to improve long read alignment. LAMSA has a substantial improvement on speed, and it is applicable to the long reads produced by the state-of-the-art platforms with various sequencing error rates. Meanwhile, it also has higher or equal base-wise sensitivity and accuracy, comparing to the state-of-the-art aligners.

On both of low- and high-error rate datasets, the numbers of the breakpoints recovered by LAMSA are higher or similar to that of the state-of-the-art aligners. Moreover, with the specifically designed non-co-linear skeleton and gap filling, LAMSA is good at coping with non-co-linear SV events, such as inversions. Overall, LAMSA is sensitive to handle the SV breakpoints of the reads.

It is also worth noting that other advanced short read alignment approaches could further improve LAMSA. For example, we tried to use the fastest mode of GEM mapper to generate the approximate matches of the fragments. With this configuration, the speed of LAMSA can be overall about 2-folds faster on the 3 low error rate and the 4 high error rate 10 000 bp simulated datasets, than that of the setting of LAMSA used in the Results Section, while it can also achieve comparable sensitivity and accuracy (Supplementary Fig. S14 and Supplementary Tables S10 and S11). This indicates us that it is also important to integrate novel sequence alignment approaches into LAMSA in the future to further improve the tool, as more advanced approaches are also developing.

It is still a little unfavorable that, some read parts cannot be confidently aligned. For example, for some reads, LAMSA aligned a portion of their bases to many genomic positions with equally high scores. This usually happens when some short read parts are from the repetitive regions of the genome, and they have two flanking SV breakpoints. In this situation, even the large read length cannot help much, since the SV breakpoints flanked read part can be seen as a standalone short sequence which must be handled independently. Considering its repetitiveness and/or serious sequencing errors, this is a hard and open problem. A possible solution could be to implement realignment based on the alignments of the multiple reads within the same local regions to filter the alignments of LAMSA. This may improve the specificity while retain the high sensitivity. However, as the post-processing also highly depends on the objective of downstream analysis, the method still needs to be specifically designed. It is an important future work for us to solve this problem to further improve the quality of the alignment.

Acknowledgements

We are very grateful to Prof. T. W. Lam and Dr S. M. Yiu in University of Hongkong for the initial discussion on this work.

Funding

This work has been partially supported by the Nature Science Foundation of China (Nos: 61301204 and 31301089), the High-Tech Research and Development Program (863) of China (Nos: 2015AA020101, 2015AA020108 and 2014AA021505).

Conflict of interest: none declared.

References

Bartenhagen,C. and Dugas,M. (2013) RSVSim: an R/Bioconductor package for the simulation of structural variations. *Bioinformatics*, **29**, 1679–1681.

Chaisson,M.J. and Tesler,G. (2012) Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC Bioinformatics*, **13**, 238.

Chaisson,M.J. *et al.* (2015) Resolving the complexity of the human genome using single-molecule sequencing. *Nature*, **517**, 608–611.

De Koning,A. *et al.* (2011) Repetitive elements may comprise over two-thirds of the human genome. *PLoS Genet.*, **7**, e1002384.

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

Dobin,A. *et al.* (2012) STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, **29**, 15–21.

Eid,J. *et al.* (2009) Real-time DNA sequencing from single polymerase molecules. *Science*, **323**, 133–138.

Eisenstein,M. (2012) Oxford Nanopore announcement sets sequencing sector abuzz. *Nat. Biotechnol.*, **30**, 295–296.

Faust,G.G. and Hall,I.M. (2012) YAHA: fast and flexible long-read alignment with optimal breakpoint detection. *Bioinformatics*, **28**, 2417–2424.

Ferragina,P. and Manzini,G. (2000) Opportunistic data structures with applications. In *Proceedings of the 41st Symposium on Foundations of Computer Science (FOCS 2000), IEEE Computer Society*, pp. 390–398.

Feuk,L. *et al.* (2006) Structural variation in the human genome. *Nat. Rev. Genet.*, **7**, 85–97.

Fonseca,N.A. *et al.* (2012) Tools for mapping high-throughput sequencing data. *Bioinformatics*, **28**, 3169–3177.

Huddleston,J. *et al.* (2014) Reconstructing complex regions of genomes using long-read sequencing technology. *Genome Res.*, **24**, 688–696.

Ip,C.L.C. *et al.* (2015) MinION Analysis and Reference Consortium: Phase 1 data release and analysis. *F1000Res.*, **4**, 1075.

Jiang,Y. *et al.* (2012) PRISM: pair-read informed split-read mapping for base-pair level detection of insertion, deletion and structural variants. *Bioinformatics*, **28**, 2576–2583.

Kent,W.J. (2002) BLAT—the BLAST-like alignment tool. *Genome Res.*, **12**, 656–664.

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022

Kiełbasa,S.M. *et al*. (2011) Adaptive seeds tame genomic sequence comparison. *Genome Res*., **21**, 487–493.

Koren,S. *et al*. (2012) Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nat. Biotechnol*., **30**, 693–700.

Langmead,B. *et al*. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol*., **10**, R25.

Langmead,B. *et al*. (2012) Fast gapped-read alignment with Bowtie 2. *Nat. Methods*, **9**, 357–359.

Li,H. (2013) Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv preprint arXiv:1303.3997*, 589–595.

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

Li,H. and Durbin,R. (2010) Fast and accurate long-read alignment with Burrows–Wheeler transform. *Bioinformatics*, **26**, 589–595.

Li,H. *et al*. (2009) The sequence alignment/map format and SAMtools. *Bioinformatics*, **25**, 2078–2079.

Li,H. *et al*. (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res*., **18**, 1851–1858.

Lim,J. *et al*. (2015) BatAlign: an incremental method for accurate alignment of sequencing reads. *Nucleic Acids Res*., **43**, e107.

Liu,C.M. *et al*. (2012) SOAP3: ultra-fast GPU-based parallel alignment tool for short reads. *Bioinformatics*, **28**, 878–879.

MacDonald,J.R. *et al*. (2014) The database of genomic variants: a curated collection of structural variation in the human genome. *Nucleic Acids Res*., **42**, D986–D992.

Marco-Sola,S. *et al*. (2012) The GEM mapper: fast, accurate and versatile alignment by filtration. *Nat. Methods*, **9**, 1185–1188.

McKenna,A. *et al*. (2010) The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res*., **20**, 1297–1303.

Mills,R.E. *et al*. (2011) Mapping copy number variation by population-scale genome sequencing. *Nature*, **470**, 59–65.

Mu,J.C. *et al*. (2012) Fast and accurate read alignment for resequencing. *Bioinformatics*, **28**, 2366–2373.

Ning,Z. *et al*. (2001) SSAHA: a fast search method for large DNA databases. *Genome Res*., **11**, 1725–1729.

Ono,Y. *et al*. (2013) PBSIM: PacBio reads simulator–toward accurate genome assembly. *Bioinformatics*, **29**, 119–121.

Schneider,G.F. and Dekker,C. (2012) DNA sequencing with nanopores. *Nat. Biotechnol*., **30**, 326–328.

Sudmant,P.H. *et al*. (2015) An integrated map of structural variation in 2,504 human genomes. *Nature*, **526**, 75–81.

Treangen,T.J. and Salzberg,S.L. (2012) Repetitive DNA and next-generation sequencing: computational challenges and solutions. *Nat. Rev. Genet*., **13**, 36–46.

Downloaded from https://academic.oup.com/bioinformatics/article/33/2/192/2525703 by guest on 16 August 2022