TECHNICAL REPORT

Minimap2: fast pairwise alignment for long nucleotide sequences

Heng Li

Broad Institute, 415 Main Street, Cambridge, MA 02142, USA

ABSTRACT

Summary: Minimap2 is a general-purpose mapper to align long noisy DNA or mRNA sequences against a large reference database. It targets query sequences of 1kb–100Mb in length with per-base divergence typically below 25%. For DNA sequence reads, minimap2 is ∼30 times faster than many mainstream long-read aligners and achieves higher accuracy on simulated data. It also employs concave gap cost and rescues inversions for improved alignment around potential structural variations. For real long RNA-seq reads, minimap2 is ∼40 times faster than peers and produces alignment more consistent with existing gene annotations.

Availability and implementation: https://github.com/lh3/minimap2

Contact: hengli@broadinstitute.org

1 INTRODUCTION

Single Molecule Real-Time (SMRT) sequencing technology and Oxford Nanopore technologies (ONT) produce reads over 10kbp in length at an error rate ∼15%. Several aligners have been developed for such data (Chaisson and Tesler, 2012; Li, 2013; Liu et al., 2016; Sović et al., 2016; Liu et al., 2017; Lin and Hsu, 2017; Sedlazeck et al., 2017). Most of them were five times as slow as mainstream short-read aligners (Langmead and Salzberg, 2012; Li, 2013) in terms of the number of bases mapped per second. We speculated there could be substantial room for speedup on the thought that 10kb long sequences should be easier to map than 100bp reads because we can more effectively skip repetitive regions, which are often the bottleneck of short-read alignment. We confirmed our speculation by achieving approximate mapping 50 times faster than BWA-MEM (Li, 2016). Suzuki (2016) extended our work with a fast and novel algorithm on generating base-level alignment, which in turn inspired us to develop minimap2 towards higher accuracy and more practical functionality.

Both SMRT and ONT have been applied to sequence spliced mRNAs (RNA-seq). While traditional mRNA aligners work (Wu and Watanabe, 2005; Iwata and Gotoh, 2012), they are not optimized for long noisy sequence reads and are tens of times slower than dedicated long-read aligners. When developing minimap2 initially for aligning genomic DNA only, we realized minor modifications could make it competitive for aligning mRNAs as well. Minimap2 is a first RNA-seq aligner specifically designed for long noisy reads.

2 METHODS

Minimap2 follows a typical seed-chain-align procedure as is used by most full-genome aligners. It collects minimizers (Roberts et al., 2004) of the reference sequences and indexes them in a hash table. Then for each query sequence, minimap2 takes query minimizers as *seeds*, finds matches to the reference, and identifies sets of colinear seeds, which are called *chains*. If base-level alignment is requested, minimap2 applies dynamic programming (DP) to extend from the ends of chains and to close unseeded regions between adjacent seeds in chains.

Minimap2 uses indexing and seeding algorithms similar to minimap (Li, 2016), and furthers the predecessor with more accurate chaining, the ability to produce base-level alignment and the support of spliced alignment.

2.1 Chaining

2.1.1 Chaining

An *anchor* is a 3-tuple (x, y, w), indicating interval $[x - w + 1, x]$ on the reference matching interval $[y - w + 1, y]$ on the query. Given a list of anchors sorted by ending reference position x, let $f(i)$ be the maximal chaining score up to the i-th anchor in the list. $f(i)$ can be calculated with dynamic programming:

$$f(i) = \max\big\{\max_{i>j\geq 1}\{f(j) + \alpha(j,i) - \beta(j,i)\}, w_i\big\} \quad (1)$$

where $\alpha(j,i) = \min\big\{\min\{y_i - y_j, x_i - x_j\}, w_i\big\}$ is the number of matching bases between the two anchors. $\beta(j,i) > 0$ is the gap cost. It equals ∞ if $y_j \geq y_i$ or $\max\{y_i - y_j, x_i - x_j\} > G$ (i.e. the distance between two anchors is too large); otherwise

$$\beta(j,i) = \gamma_c\big((y_i - y_j) - (x_i - x_j)\big) \quad (2)$$

In implementation, a gap of length l costs $\gamma_c(l) = 0.01 \cdot \bar{w} \cdot |l| + 0.5 \log_2 |l|$, where $\bar{w}$ is the average seed length. For m anchors, directly computing all $f(\cdot)$ with Eq. (1) takes $O(m^2)$ time. Although theoretically faster chaining algorithms exist (Abouelhoda and Ohlebusch, 2005), they are inapplicable to generic gap cost, complex to implement and usually associated with a large constant. We introduced a simple heuristic to accelerate chaining.

We note that if anchor i is chained to j, chaining i to a predecessor of j is likely to yield a lower score. When evaluating Eq. (1), we start from anchor $i - 1$ and stop the process if we cannot find a better score after up to h iterations. This approach reduces the average time to $O(h \cdot m)$. In practice, we can almost always find the optimal chain with $h = 50$; even if the heuristic fails, the optimal chain is often close.

2.1.2 Backtracking

Let $P(i)$ be the index of the best predecessor of anchor i. It equals 0 if $f(i) = w_i$ or $\text{argmax}_j\{f(j) + \eta(j,i) - \gamma(j,i)\}$ otherwise. For each anchor i in the descending order of $f(i)$, we apply $P(\cdot)$ repeatedly to find its predecessor and mark each visited i as 'used', until $P(i) = 0$ or we reach an already 'used' i. This way we find all chains with no anchors used in more than one chains.

2.1.3 Identifying primary chains

In the absence of copy number changes, each query segment should not be mapped to two places in the reference. However, chains found at the previous step may have significant or complete overlaps due to repeats in the reference. Minimap2 used the following procedure to identify *primary chains* that do not greatly overlap on the query. Let Q be an empty set initially. For each chain from the best to the worst according to their chaining scores: if on the query, the chain overlaps with a chain in Q by 50% or higher percentage of the shorter chain, mark the chain as secondary to the chain in Q; otherwise, add the chain to Q. In the end, Q contains all the primary chains. We did not choose a more sophisticated data structure (e.g. range tree or k-d tree) because this step is not the performance bottleneck.

2.2 Aligning genomic DNA

2.2.1 Alignment with 2-piece affine gap cost

Minimap2 performs DP-based global alignment between adjacent anchors in a chain. It uses a 2-piece affine gap cost (Gotoh, 1990):

$$\gamma_a(l) = \min\{q + |l| \cdot e, \tilde{q} + |l| \cdot \tilde{e}\} \tag{3}$$

Without losing generality, we always assume $q+e < \tilde{q}+\tilde{e}$. On the condition that $e > \tilde{e}$, it applies cost $q + |l| \cdot e$ to gaps shorter than $\lceil(\tilde{q} - q)/(e - \tilde{e})\rceil$ and applies $\tilde{q} + |l| \cdot \tilde{e}$ to longer gaps. This scheme helps to recover longer insertions and deletions (INDELs).

The equation to compute the optimal alignment under $\gamma_a(\cdot)$ is

$$\left\{\begin{array}{l} H_{ij} = \max\{H_{i-1,j-1} + s(i,j), E_{ij}, F_{ij}, \tilde{E}_{ij}, \tilde{F}_{ij}\} \\ E_{i+1,j} = \max\{H_{ij} - q, E_{ij}\} - e \\ F_{i,j+1} = \max\{H_{ij} - q, F_{ij}\} - e \\ \tilde{E}_{i+1,j} = \max\{H_{ij} - \tilde{q}, \tilde{E}_{ij}\} - \tilde{e} \\ \tilde{F}_{i,j+1} = \max\{H_{ij} - \tilde{q}, \tilde{F}_{ij}\} - \tilde{e} \end{array}\right. \tag{4}$$

where $s(i,j)$ is the score between the i-th reference base and j-th query base. Eq. (4) is a natural extension to the equation under affine gap cost (Gotoh, 1982; Altschul and Erickson, 1986).

2.2.2 Suzuki's formulation

When we allow gaps longer than several hundred base pairs, nucleotide-level alignment is much slower than chaining. SSE acceleration is critical to the performance of minimap2. Traditional SSE implementations (Farrar, 2007) based on Eq. (4) can achieve 16-way parallelization for short sequences, but only 4-way parallelization when the peak alignment score reaches 32767. Long sequence alignment may exceed this threshold. Inspired by Wu et al. (1996) and the following work, Suzuki (2016) proposed a difference-based formulation that lifted this limitation. In case of 2-piece gap cost, define

$$\left\{\begin{array}{ll} u_{ij} \triangleq H_{ij} - H_{i-1,j} & v_{ij} \triangleq H_{ij} - H_{i,j-1} \\ x_{ij} \triangleq E_{i+1,j} - H_{ij} & \tilde{x}_{ij} \triangleq \tilde{E}_{i+1,j} - \tilde{H}_{ij} \\ y_{ij} \triangleq F_{i,j+1} - H_{ij} & \tilde{y}_{ij} \triangleq \tilde{F}_{i,j+1} - \tilde{H}_{ij} \end{array}\right.$$

We can transform Eq. (4) to

$$\left\{\begin{array}{lcl} z_{ij} & = & \max\{s(i,j), x_{i-1,j} + v_{i-1,j}, y_{i,j-1} + u_{i,j-1}, \\ & & \tilde{x}_{i-1,j} + v_{i-1,j}, \tilde{y}_{i,j-1} + u_{i,j-1}\} \\ u_{ij} & = & z_{ij} - v_{i-1,j} \\ v_{ij} & = & z_{ij} - u_{i,j-1} \\ x_{ij} & = & \max\{0, x_{i-1,j} + v_{i-1,j} - z_{ij} + q\} - q - e \\ y_{ij} & = & \max\{0, y_{i,j-1} + u_{i,j-1} - z_{ij} + q\} - q - e \\ \tilde{x}_{ij} & = & \max\{0, \tilde{x}_{i-1,j} + v_{i-1,j} - z_{ij} + \tilde{q}\} - \tilde{q} - \tilde{e} \\ \tilde{y}_{ij} & = & \max\{0, \tilde{y}_{i,j-1} + u_{i,j-1} - z_{ij} + \tilde{q}\} - \tilde{q} - \tilde{e} \end{array}\right. \tag{5}$$

where z_{ij} is a temporary variable that does not need to be stored.

An important property of Eq. (5) is that all values are bounded by scoring parameters. To see that,

$$x_{ij} = E_{i+1,j} - H_{ij} = \max\{-q, E_{ij} - H_{ij}\} - e$$

With $E_{ij} \le H_{ij}$, we have

$$-q - e \le x_{ij} \le \max\{-q, 0\} - e = -e$$

and similar inequations for y_{ij}, $\tilde{x}_{ij}$ and $\tilde{y}_{ij}$. In addition,

$$u_{ij} = z_{ij} - v_{i-1,j} \ge \max\{x_{i-1,j}, \tilde{x}_{i-1,j}\} \ge -q - e$$

As the maximum value of $z_{ij} = H_{ij} - H_{i-1,j-1}$ is M, the maximal matching score, we can derive

$$u_{ij} \le M - v_{i-1,j} \le M + q + e$$

In conclusion, in Eq. (5), x and y are bounded by $[-q-e, -e]$, $\tilde{x}$ and $\tilde{y}$ by $[-\tilde{q}-\tilde{e}, -\tilde{e}]$, and u and v by $[-q-e, M+q+e]$. When $-128 \le -q-e < M+q+e \le 127$, each of them can be stored as a 8-bit integer. This enables 16-way SSE vectorization regardless of the peak score of the alignment.

For a more efficient SSE implementation, we transform the row-column coordinate to the diagonal-antidiagonal coordinate by letting $r \leftarrow i + j$ and $t \leftarrow i$. Eq. (5) becomes:

$$\left\{\begin{array}{lcl} z_{rt} & = & \max\{s(t, r-t), x_{r-1,t-1} + v_{r-1,t-1}, y_{r-1,t} \\ & & +u_{r-1,t}, \tilde{x}_{r-1,t-1} + v_{r-1,t-1}, \tilde{y}_{r-1,t} + u_{r-1,t}\} \\ u_{rt} & = & z_{rt} - v_{r-1,t-1} \\ v_{rt} & = & z_{rt} - u_{r-1,t} \\ x_{rt} & = & \max\{0, x_{r-1,t-1} + v_{r-1,t-1} - z_{rt} + q\} - q - e \\ y_{rt} & = & \max\{0, y_{r-1,t} + u_{r-1,t} - z_{rt} + q\} - q - e \\ \tilde{x}_{rt} & = & \max\{0, \tilde{x}_{r-1,t-1} + v_{r-1,t-1} - z_{rt} + \tilde{q}\} - \tilde{q} - \tilde{e} \\ \tilde{y}_{rt} & = & \max\{0, \tilde{y}_{r-1,t} + u_{r-1,t} - z_{rt} + \tilde{q}\} - \tilde{q} - \tilde{e} \end{array}\right.$$

In this formulation, cells with the same diagonal index r are independent of each other. This allows us to fully vectorize the computation of all cells on the same anti-diagonal in one inner loop. It also simplifies banded alignment, which would be difficult with striped vectorization (Farrar, 2007).

On the condition that $q + e < \tilde{q} + \tilde{e}$ and $e > \tilde{e}$, the initial values in the diagonal-antidiagonal formuation is

$$\left\{\begin{array}{l} x_{r-1,-1} = y_{r-1,r} = -q - e \\ \tilde{x}_{r-1,-1} = \tilde{y}_{r-1,r} = -\tilde{q} - \tilde{e} \\ u_{r-1,r} = v_{r-1,-1} = \eta(r) \end{array}\right.$$

where

$$\eta(r) = \left\{\begin{array}{ll} -q - e & (r = 0) \\ -e & (r < \lceil\frac{\tilde{q}-q}{e-\tilde{e}} - 1\rceil) \\ r \cdot (e - \tilde{e}) - (\tilde{q} - q) - \tilde{e} & (r = \lceil\frac{\tilde{q}-q}{e-\tilde{e}} - 1\rceil) \\ -\tilde{e} & (r > \lceil\frac{\tilde{q}-q}{e-\tilde{e}} - 1\rceil) \end{array}\right.$$

These can be derived from the initial values for Eq. (4).

In practice, our 16-way vectorized implementation of global alignment is three times as fast as Parasail's 4-way vectorization (Daily, 2016). Without banding, our implementation is slower than Edlib (Šošić and Šikic, 2017), but with a 1000bp band, it is considerably faster. When performing global alignment between anchors, we expect the alignment to stay close to the diagonal of the DP matrix. Banding is applicable most of time.

2.2.3 The Z-drop heuristic

With global alignment, minimap2 may force to align unrelated sequences between two adjacent anchors. To avoid such an artifact, we compute accumulative alignment score along the alignment path and break the alignment where the score drops too fast in the diagonal direction. More precisely, let $S(i,j)$ be the alignment score along the alignment path ending at cell (i,j) in the DP matrix. We break the alignment if there exist (i',j') and (i,j), $i' < i$ and $j' < j$, such that

$$S(i',j') - S(i,j) > Z + e \cdot |(i - i') - (j - j')|$$

where e is the gap extension cost and Z is an arbitrary threshold. This strategy is first used in BWA-MEM. It is similar to X-drop employed in BLAST (Altschul et al., 1997), but unlike X-drop, it would not break the alignment in the presence of a single long gap.

When minimap2 breaks a global alignment between two anchors, it performs local alignment between the two subsequences involved in the global alignment, but this time with the one subsequence reverse complemented. This additional alignment step may identify short inversions that are missed during chaining.

2.3 Aligning spliced sequences

The algorithm described above can be adapted to spliced alignment. In this mode, the chaining gap cost distinguishes insertions to and deletions from the reference: $\gamma_c(l)$ in Eq. (2) takes the form of

$$\gamma_c(l) = \left\{\begin{array}{ll} 0.01 \cdot \bar{w} \cdot l + 0.5 \log_2 l & (l > 0) \\ \min\{0.01 \cdot \bar{w} \cdot |l|, \log_2 |l|\} & (l < 0) \end{array}\right.$$

Similarly, the gap cost function used for DP-based alignment is changed to

$$\gamma_a(l) = \left\{\begin{array}{ll} q + l \cdot e & (l > 0) \\ \min\{q + |l| \cdot e, \tilde{q}\} & (l < 0) \end{array}\right.$$

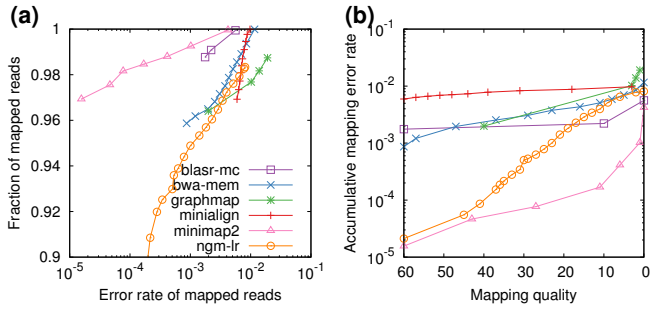

Fig. 1. Evaluation on simulated SMRT reads aligned against human genome GRCh38. 33,088 ≥1000bp reads were simulated using pbsim (Ono et al., 2013) with error profile sampled from file 'm131017_060208_42213_*.1.*' downloaded at http://bit.ly/chm1p5c3. The N50 read length is 11,628. A read is considered correctly mapped if the true position overlaps with the best mapping position by 10% of the read length. All aligners were run under the default setting for SMRT reads. (a) ROC-like curve. Alignments are sorted by mapping quality in the descending order. For each mapping quality threshold, the fraction of alignments with mapping quality above the threshold and their error rate are plotted. Kart outputted all alignments at mapping quality 60, so is not shown in the figure. It mapped nearly all reads with 4.1% of alignments being wrong, less accurate than others. (b) Accumulative mapping error rate as a function of mapping quality.

In alignment, a deletion no shorter than $\lceil(\tilde{q}-q)/e\rceil$ is regarded as an intron, which pays no cost to gap extensions.

To pinpoint precise splicing junctions, minimap2 introduces reference-dependent cost to penalize non-canonical splicing:

$$\begin{cases} H_{ij} = \max\{H_{i-1,j-1} + s(i,j), E_{ij}, F_{ij}, \tilde{E}_{ij} - a(i)\} \\ E_{i+1,j} = \max\{H_{ij} - q, E_{ij}\} - e \\ F_{i,j+1} = \max\{H_{ij} - q, F_{ij}\} - e \\ \tilde{E}_{i+1,j} = \max\{H_{ij} - d(i) - \tilde{q}, \tilde{E}_{ij}\} \end{cases} \quad (6)$$

Let T be the reference sequence. $d(i)$ is the cost of a non-canonical donor site, which takes 0 if $T[i+1,i+2] =$ `GT`, or a positive number p otherwise. Similarly, $a(i)$ is the cost of a non-canonical acceptor site, which takes 0 if $T[i-1,i] =$ `AG`, or p otherwise. Eq. (6) is almost equivalent to the equation used by EXALIN (Zhang and Gish, 2006) except that we allow insertions immediately followed by deletions and vice versa; in addition, we use Suzuki's diagonal formulation in actual implementation.

If RNA-seq reads are not sequenced from stranded libraries, the read strand relative to the underlying transcript is unknown. By default, minimap2 aligns each chain twice, first assuming `GT`–`AG` as the splicing signal and then assuming `CT`–`AC`, the reverse complement of `GT`–`AG`, as the splicing signal. The alignment with a higher score is taken as the final alignment. This procedure also infers the relative strand of reads that span canonical splicing sites.

In the spliced alignment mode, minimap2 further increases the density of minimizers and disables banded alignment. Together with the two-round DP-based alignment, spliced alignment is several times slower than DNA sequence alignment.

3 RESULTS

3.1 Aligning genomic reads

As a sanity check, we evaluated minimap2 on simulated human reads along with BLASR (v1.MC.rc64; Chaisson and Tesler, 2012), BWA-MEM (v0.7.15; Li, 2013), GraphMap (v0.5.2; Sović et al., 2016), Kart (v2.2.5; Lin and Hsu, 2017), minialign (v0.5.3; Suzuki, 2016) and NGMLR (v0.2.5; Sedlazeck et al., 2017). We excluded rHAT (Liu et al., 2016) and LAMSA (Liu et al., 2017) because they either crashed or produced malformatted output. In this evaluation, minimap2 has higher power to distinguish unique and repetitive hits, and achieves overall higher mapping accuracy (Fig. 1a). It is still the most accurate even if we skip DP-based alignment (data not shown), confirming chaining alone is sufficient to achieve high accuracy for approximate mapping. Minimap2 and NGMLR provide better mapping quality estimate: they rarely give repetitive hits high mapping quality (Fig. 1b). Apparently, other aligners may occasionally miss close suboptimal hits and be overconfident in wrong mappings. On run time, minialign is slightly faster than minimap2 and Kart. They are over 30 times faster than the rest. Minimap2 consumed 6.1GB memory at the peak, more than BWA-MEM but less than others.

On real human SMRT reads, the relative performance and sensitivity of these aligners are broadly similar to the metrics on simulated data. We are unable to provide a good estimate of mapping error rate due to the lack of the truth. On ONT ∼100kb human reads (Jain et al., 2017), BWA-MEM failed. Kart, minialign and minimap2 are over 70 times faster than others. We have also examined tens of ≥100bp INDELs in IGV (Robinson et al., 2011) and can confirm the observation by Sedlazeck et al. (2017) that BWA-MEM often breaks them into shorter gaps. The issue is much alleviated with minimap2, thanks to the 2-piece affine gap cost.

3.2 Aligning spliced reads

We evaluated minimap2 on SIRV control data (AC:SRR5286959; Byrne et al., 2017) where the truth is known. Minimap2 predicted 59 916 introns from 11 017 reads. 93.0% of splice junctions are precise. We examined wrongly predicted junctions and found the majority were caused by clustered splicing signals (e.g. two adjacent `GT` sites). When INDEL sequencing errors are frequent, it is difficult to find precise splicing sites in this case. If we allow up to 10bp distance from true splicing sites, 98.4% of aligned introns are approximately correct. Given this observation, we might be able to improve boundary detection by initializing $d(\cdot)$ and $a(\cdot)$ in Eq. (6) with position-specific scoring matrices or more sophisticated models. We have not tried this approach.

We next aligned real mouse reads (Byrne et al., 2017) with GMAP (v2017-06-20; Wu and Watanabe, 2005), minimap2, SpAln (v2.3.1; Iwata and Gotoh, 2012) and STAR (v2.5.3a; Dobin et al., 2013). In general, minimap2 is more consistent with existing annotations (Table 1): it finds more junctions with a higher percentage being exactly or approximately correct. Minimap2 is over 40 times faster than GMAP and SpAln. While STAR is close to minimap2 in speed, it does not work well with noisy reads. We have also evaluated spliced aligners on public Iso-Seq data (human Alzheimer brain from http://bit.ly/isoseqpub). The observation is similar: minimap2 is faster at higher junction accuracy.

We noted that GMAP and SpAln have not been optimized for noisy reads. We are showing the best setting we have experimented, but their developers should be able to improve their accuracy further.

4 CONCLUSION

Minimap2 is a fast, accurate and versatile aligner for long nucleotide sequences. In addition to reference-based read mapping, minimap2

Table 1. Evaluation of junction accuracy on 2D ONT reads

	GMAP	minimap2	SpAln	STAR
Run time (CPU min)	631	15.5	2 076	33.9
Peak RAM (GByte)	8.9	14.5	3.2	29.2
# aligned reads	103 669	103 917	103 711	26 479
# chimeric alignments	1 904	1 671	0	0
# non-spliced alignments	15 854	14 483	17 033	10 545
# aligned introns	692 275	694 237	692 945	78 603
# novel introns	11 239	3 217	8 550	1 214
% exact introns	83.8%	91.8%	87.9%	55.2%
% approx. introns	91.8%	96.5%	92.5%	82.4%

Mouse reads (AC:SRR5286960) were mapped to the primary assembly of mouse genome GRCm38 with the following tools and command options: minimap2 ('-ax splice'); GMAP ('-n 0 –min-intronlength 30 –cross-species'); SpAln ('-Q7 -LS -S3'); STARlong (according to http://bit.ly/star-pb). The alignments were compared to the EnsEMBL gene annotation, release 89. A predicted intron is *novel* if it has no overlaps with any annotated introns. An intron is *exact* if it is identical to an annotated intron. An intron is *approximate* if both its 5'- and 3'-end are within 10bp around the ends of an annotated intron.

inherits minimap's functionality to search against huge multi-species databases and to find read overlaps. On a few test data sets, minimap2 appears to yield slightly better miniasm assembly (Li, 2016). Minimap2 can also align similar genomes or different assemblies of the same species. However, full-genome alignment is an intricate research topic. More thorough evaluations would be necessary to justify the use of minimap2 for such applications.

ACKNOWLEDGEMENTS

We owe a debt of gratitude to Hajime Suzuki for releasing his masterpiece and insightful notes before formal publication. We thank M. Schatz, P. Rescheneder and F. Sedlazeck for pointing out the limitation of BWA-MEM. We are also grateful to early minimap2 testers who have greatly helped to suggest features and to fix various issues.

REFERENCES

Abouelhoda, M. I. and Ohlebusch, E. (2005). Chaining algorithms for multiple genome comparison. *J. Discrete Algorithms*, 3:321–41.

Altschul, S. F. and Erickson, B. W. (1986). Optimal sequence alignment using affine gap costs. *Bull Math Biol*, 48:603–16.

Altschul, S. F. et al. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25:3389–402.

Byrne, A. et al. (2017). Nanopore long-read RNAseq reveals widespread transcriptional variation among the surface receptors of individual B cells. *Nat Commun*, 8:16027.

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.

Daily, J. (2016). Parasail: SIMD C library for global, semi-global, and local pairwise sequence alignments. *BMC Bioinformatics*, 17:81.

Dobin, A. et al. (2013). STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29:15–21.

Farrar, M. (2007). Striped Smith-Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics*, 23:156–61.

Gotoh, O. (1982). An improved algorithm for matching biological sequences. *J Mol Biol*, 162:705–8.

Gotoh, O. (1990). Optimal sequence alignment allowing for long gaps. *Bull Math Biol*, 52:359–73.

Iwata, H. and Gotoh, O. (2012). Benchmarking spliced alignment programs including Spaln2, an extended version of Spaln that incorporates additional species-specific features. *Nucleic Acids Res*, 40:e161.

Jain, M. et al. (2017). Nanopore sequencing and assembly of a human genome with ultra-long reads. *bioRxiv*. doi:10.1101/128835.

Langmead, B. and Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. *Nat Methods*, 9:357–9.

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

Li, H. (2016). Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics*, 32:2103–10.

Lin, H.-N. and Hsu, W.-L. (2017). Kart: a divide-and-conquer algorithm for NGS read alignment. *Bioinformatics*.

Liu, B. et al. (2016). rHAT: fast alignment of noisy long reads with regional hashing. *Bioinformatics*, 32:1625–31.

Liu, B. et al. (2017). LAMSA: fast split read alignment with long approximate matches. *Bioinformatics*, 33:192–201.

Ono, Y. et al. (2013). PBSIM: PacBio reads simulator–toward accurate genome assembly. *Bioinformatics*, 29:119–21.

Roberts, M. et al. (2004). Reducing storage requirements for biological sequence comparison. *Bioinformatics*, 20:3363–9.

Robinson, J. T. et al. (2011). Integrative genomics viewer. *Nat Biotechnol*, 29:24–6.

Sedlazeck, F. J. et al. (2017). Accurate detection of complex structural variations using single molecule sequencing. *bioRxiv*. doi:10.1101/169557.

Šošić, M. and Šikic, M. (2017). Edlib: a C/C++ library for fast, exact sequence alignment using edit distance. *Bioinformatics*, 33:1394–1395.

Sović, I. et al. (2016). Fast and sensitive mapping of nanopore sequencing reads with GraphMap. *Nat Commun*, 7:11307.

Suzuki, H. (2016). Fast and accurate alignment tool for pacbio and nanopore long reads. https://github.com/ocxtal/minialign.

Wu, S. et al. (1996). A subquadratic algorithm for approximate limited expression matching. *Algorithmica*, 15:50–67.

Wu, T. D. and Watanabe, C. K. (2005). GMAP: a genomic mapping and alignment program for mRNA and EST sequences. *Bioinformatics*, 21:1859–75.

Zhang, M. and Gish, W. (2006). Improved spliced alignment from an information theoretic approach. *Bioinformatics*, 22:13–20.