


MECAT: fast mapping, error correction, and *de novo* assembly for single-molecule sequencing reads

Chuan-Le Xiao^{1,4,8} , Ying Chen^{2,8}, Shang-Qian Xie^{1,5} , Kai-Ning Chen¹, Yan Wang¹, Yue Han^{1,6}, Feng Luo⁷ , & Zhi Xie¹ 

We present a tool that combines fast mapping, error correction, and *de novo* assembly (MECAT; accessible at <https://github.com/xiaochuanle/MECAT>) for processing single-molecule sequencing (SMS) reads. MECAT's computing efficiency is superior to that of current tools, while the results MECAT produces are comparable or improved. MECAT enables reference mapping or *de novo* assembly of large genomes using SMS reads on a single computer.

SMS technologies¹ developed by companies such as Pacific Bioscience^{2,3} and Oxford Nanopore^{4,5} yield long reads that have many advantages in genomics studies^{3,6–9}. Although SMS is widely used in genomics of small bacteria and archaea³, the application of SMS to mid- or large-sized genomes^{8–10} has incurred high computational cost^{11,12}.

The pairwise and reference genome alignment steps are very computationally costly for SMS reads¹¹. Usually, the k-mer match-based method is used to first filter out random read pairs and quickly find seed alignments. However, because of the highly repetitive nature of biology genomes¹³, reads sampled from repetitive regions can lead to a large number of k-mer matches¹⁴, which results in excessive candidate matches¹³. Simply masking low-complexity sequences, or ignoring the highly repetitive k-mer matches, can cause loss of correct overlaps¹⁴. Thus, local alignments are needed to find well-matched reads or best matched genome locations¹⁵. In BLASR¹⁵, the best arrangement of k-mer pairs is solved by slow sparse dynamic programming. Even with a fast linear local alignment program, such as diff¹⁶

in DALIGNER¹⁷, the computational cost for local alignments between two long SMS reads, or between an SMS read and a reference genome, is still high^{16,17}. The local alignment of excessive candidate matches takes up to 70% computational time in pairwise and reference genome alignment of SMS reads. Recently, the Canu¹⁴ pipeline employed a term frequency–inverse document frequency (tf-idf) k-mer-weighting method to reduce the effects of repetitive k-mer matches. However, Canu did not consider the arrangement of k-mer pairs. Therefore, there are still many excessive matches.

Meanwhile, many SMS applications—such as SMS read correction and genome assembly—need only a limited number of matched reads^{15,18,19}. Because of the repetitive nature of genomes, the number of matched k-mer pairs does not correspond to the overlapping lengths and so cannot be used as the criteria to directly select high-quality, reliable matches. Local alignments are needed to screen a large number of candidate matches, which dramatically increases the computational cost of SMS read correction and genome assembly.

Here, we developed a pseudolinear alignment scoring algorithm to filter excessive alignments (Fig. 1 a–e). Our algorithm uses the distance difference factors (DDFs) to score matched k-mer pairs in two steps (see Online Methods). The score of the seed k-mer pair is supported by all matched k-mer pairs and their interval distance. Thus, the scores represent the global matching information between two SMS reads, or between an SMS read and the reference genome. The scores of seed k-mer pairs between the read pairs grow linearly with their overlapping lengths in PacBio data from four different genomes¹¹ (Fig. 1f). Therefore, by selecting SMS read pairs with high scores, we can filter out noninformative candidate alignments. After filtering by DDF scoring, we reduced candidate alignments by 50% to 70% before proceeding with further local alignment using diff (Fig. 1g), which made the aligner 2–3× faster than those without DDF score filtering.

Based on our DDF alignment scoring algorithm, we developed a fast aligner named MECAT, which can be run with or without local alignment. We first evaluated the performance of MECAT aligner on pairwise alignment. We compared MECAT aligner to two SMS read pairwise alignment tools, MHAP(v2.12)¹¹ and DALIGNER¹⁷ in FALCON (v0.40)¹². For five PacBio data sets, MECAT aligner with local alignment is faster than both MHAP and DALIGNER (Table 1). For the PacBio data of large human genome, MECAT aligner is 5× faster than MHAP-fast, and 17× faster than DALIGNER. For three Nanopore data sets, MECAT aligner with local alignment is faster than both MHAP-fast and

¹State Key Laboratory of Ophthalmology, Zhongshan Ophthalmic Center, Sun Yat-sen University, Guangzhou, China. ²School of Data and Computer Science, and Guangdong Provincial Key Laboratory of Computational Science, Sun Yat-sen University, Guangzhou, China. ³Southern Regional Collaborative Innovation Center for Grain and Oil Crops in China, Hunan Agricultural University, Hunan, China. ⁴College of Plant Protection, Hunan Agricultural University, Changsha, China. ⁵Institute of Tropical Agriculture and Forestry, Hainan University, Haikou, China. ⁶CookGene Bio-technology Co., Ltd., Guangzhou, China. ⁷School of Computing, Clemson University, Clemson, South Carolina, USA. ⁸These authors contributed equally to this work. Correspondence should be addressed to C.-L.X. (xiaochuanle@126.com), F.L. (luofeng@clemson.edu), or Z.X. (xiezhi@gmail.com).

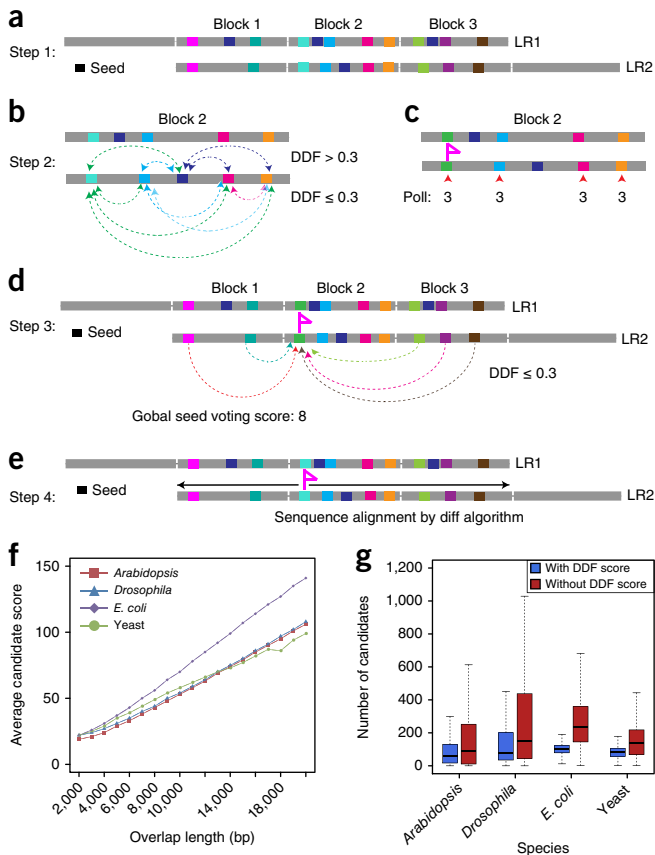


Figure 1 | Principle and property of DDF scoring algorithm in MECAT alignment. (a) Alignment of k-mers between the blocks of two SMS reads. LR, long read. (b) Pairwise scoring, using DDF, between k-mer pairs in each block pair (Block 2 in a is provided as an example). (c) Selecting the seed k-mer pair with the highest score (indicated by pink flag). We randomly select one seed pair if multiple k-mer pairs have the same score. (d) Scoring the seed k-mer pair using k-mer pairs in other block pairs. (e) Aligning two reads from the seed k-mer pair. (f) The relationship between the overlap length of two reads and their DDF scores from the four SMRT data sets (*E. coli*, yeast, *A. thaliana*, and *D. melanogaster*). (g) Comparison of the numbers of alignment candidates with and without filtering with the DDF score. The box plots the lower quartile (Q1), median (*m* or Q2), and upper quartile (Q3) of numbers of alignment candidates. The top whiskers indicate the maximum value, and the bottom whiskers indicate the minimum value.

MHAP-sensitive, but it is slower than DALIGNER (Table 1). On account of the high error rate of Nanopore data, we lowered the threshold in MECAT to obtain enough candidate matches for later error correction, and this slowed MECAT. Meanwhile, MECAT aligner without local alignment was much faster than other aligners for both the PacBio and Nanopore data sets (Table 1). Another important benefit of our DDF alignment score is that we can select reliably matched reads for a given read template based on DDF scores only. Thus, we can omit the local alignment step when only top candidate matches are needed, and this can significantly reduce computational cost for SMS applications. In addition, MECAT used a similar amount of memory as DALIGNER but much less than MHAP (Supplementary Table 1).

We evaluated the sensitivity and precision of pairwise alignment of the aligners using three simulated PacBio data sets of *Escherichia coli*, yeast, and human chr1 (Supplementary Notes 1

and 2 and Supplementary Table 2). Since the starting and ending positions of each simulated read in the reference genomes were known, we could calculate the true pairwise overlap relationships between all the reads. The sensitivity of DALIGNER¹⁷ is the best among the four aligners, but its precision is the lowest. The precision and sensitivity of DALIGNER became highly unbalanced for the human chr1 data set (9.1% precision). Conversely, MHAP¹¹ has high precision but low sensitivity. The sensitivity of MECAT aligner is consistently higher than that of MHAP, while similar precision is maintained. Compared to DALIGNER, MECAT aligner has higher precision but lower sensitivity. MECAT aligner achieved a good balance between sensitivity and precision for both small and large genomes.

The DDF alignment score is sensitive to the overlap length between read and reference genome; thus, MECAT aligner is also suitable for aligning SMS reads to a reference genome. We compared MECAT aligner to BLASR (v1.3.1.142244)¹⁵ in SMRT analysis (v2.30) and BWA-mem (v0.7.12-r1044)¹⁸ for reference genome alignment (Supplementary Note 3). For four PacBio data sets of small genomes (*E. coli*, yeast, *Arabidopsis thaliana* and *Drosophila melanogaster*), MECAT aligner was 35–65× faster than BLASR and 18–70× faster than BWA-mem (Table 1). For the PacBio human genome data set, MECAT aligner was 12× faster than BLASR and 4× faster than BWA-mem. For three Nanopore data sets of small genomes (*E. coli*, *Bacillus anthracis*, and *Yersinia pestis*), MECAT was 2–5× faster than BLASR and four to 6× faster than BWA-mem. The mapping overlap rates of the three algorithms were as high as 95–99% for the same alignment positions (Supplementary Note 3 and Supplementary Fig. 1), which showed the high confidence of MECAT aligner. We compared the sensitivity, precision, and coverage of the aligners using 20× simulated PacBio data sets of *E. coli*, yeast, and human genomes (Supplementary Table 3). Compared with BLASR¹⁵ and BWA-mem¹⁸, MECAT aligner mapped a slightly lower number of reads to the reference genome, but it mapped more reads correctly for all three data sets. MECAT also has similar read coverage at regions with large structural variants (Supplementary Note 4 and Supplementary Table 4). MECAT aligner can rapidly align the SMS reads to the reference genome while maintaining high sensitivity, precision, and coverage.

The high-error SMS reads must be corrected before they are used in other applications. Corrected reads are usually constructed from consensus of a number of matched reads. The MECAT aligner allows us to quickly select candidate reads without local alignment. We developed a fast error correction tool in MECAT by using our fast aligner (see Online Methods). Experiments showed that the correcting speed of MECAT was 4–10× higher than those of FC_Consensus¹⁴, and 5–21× higher than those of FalconSense¹² for four PacBio data sets. For three Nanopore data sets, the correcting speed of MECAT was 1.06–7× higher than those of FC_Consensus and 1.6–11× higher than those of FalconSense. Furthermore, MECAT obtained higher correction accuracies for most data sets (Supplementary Note 5 and Supplementary Table 5).

Because the DDF alignment scores are correlated with the overlap size between the two reads, we were able to replace the slow overlapInCore in Canu (v1.0) with MECAT aligner to develop a fast *de novo* assembly pipeline. The MECAT aligner significantly reduced the computational time for contig construction. The

Table 1 | Computing performance of alignment of SMS reads

Data set	Pairwise alignment time (core h)					Reference alignment time (core h)		
	MHAP (Fast)	MHAP (Sensitive)	Daligner	MECAT	MECAT(L)	BLASR	BWA	MECAT(L)
<i>E. coli</i>	0.58	1.38	0.78	0.21	0.55	2.71	1.56	0.04
Yeast	1.26	4.65	2.9	0.27	0.48	12.61	6.81	0.36
<i>A. Thaliana</i>	0.79	2.11	2.88	0.26	0.43	167.45	154.89	2.45
<i>D. melanogaster</i>	0.76	1.77	2.89	0.26	0.38	160.42	216.3	3.08
Human	1.36	3.89	2.91	0.23	0.24	7184	2,511.33	553.21
<i>E. coli</i> *	1.44	1.57	0.13	0.11	0.56	0.25	0.28	0.05
<i>B. anthracis</i> *	2.63	3.05	2.52	0.39	2.10	0.90	1.48	0.22
<i>Y. pestis</i> *	1.05	1.47	0.35	0.17	0.50	0.34	0.68	0.14

*Denotes the Nanopore reads. MECAT denotes aligner without local alignment, and MECAT(L) denotes aligner with local alignment. The reported time include both index construction and alignment time. Bold font indicates best performance.

reductions of computational costs in overlapping, error correction, and contig construction steps enabled the MECAT *de novo* assembler to reconstruct the human CHM1 genome in 7,737 central processing unit (CPU) hours, which is 24.9× faster than PBcR-MHAP-fast¹¹, 56.3× faster than PBcR-MHAP-sensitive¹¹ and 5.1× faster than the Canu (v1.3)¹⁴ (**Supplementary Notes 6 and 7 and Supplementary Table 6**). We also used MECAT to assemble a diploid Han Chinese genome from 102× PacBio sequencing reads on a 32-core computer in 25 d (**Supplementary Notes 8 and Supplementary Table 7**). MECAT produced reference-quality assemblies using both PacBio and Nanopore reads (**Supplementary Notes 6–8**).

METHODS

Methods, including statements of data availability and any associated accession codes and references, are available in the [online version of the paper](#).

Note: Any Supplementary Information and Source Data files are available in the [online version of the paper](#).

ACKNOWLEDGMENTS

We thank D.P. Wang for supplying the Chinese human data set. We thank the NCBI assembly group for the Han-1 Chinese annotation. This work was collectively supported by the National Natural Science Foundation of China (31471232, 31471789 and 31600667), the Fundamental Research Funds for the Central Universities (15ykjc23d), the Guangdong Natural Science Foundation (2015A030313127), the Joint Research Fund for the Overseas Natural Science of China (3030901001222), infrastructure support from Center for Precision Medicine (Sun Yat-sen University), China Postdoctoral Science Foundation (2017M612798), and the National Institute of Food and Agriculture (NIFA), USA (2017-70016-26051).

AUTHOR CONTRIBUTIONS

C.-L.X. conceived and designed this project. Y.C. and C.-L.X. implemented the algorithms. S.-Q.X., C.L.-X., and Y.C. performed the test experiments. K.-N.C., Y.W., and Y.H. coordinated the data release and assisted with executing the pipeline. F.L. provided theoretical analysis of the algorithms. C.-L.X., F.L., Z.X., Y.C., and S.-Q.X. wrote the manuscript. All authors read and approved the final version of the manuscript.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial interests.

Reprints and permissions information is available online at <http://www.nature.com/reprints/index.html>. Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

- Schadt, E.E., Turner, S. & Kasarskis, A. *Hum. Mol. Genet.* **19**, R227–R240 (2010).
- Eid, J. *et al. Science* **323**, 133–138 (2009).
- Chin, C.S. *et al. Nat. Methods* **10**, 563–569 (2013).
- Jain, M. *et al. Nat. Methods* **12**, 351–356 (2015).
- Sović, I. *et al. Nat. Commun.* **7**, 11307 (2016).
- Loman, N.J., Quick, J. & Simpson, J.T. *Nat. Methods* **12**, 733–735 (2015).
- Koren, S. *et al. Nat. Biotechnol.* **30**, 693–700 (2012).
- Seo, J.S. *et al. Nature* **538**, 243–247 (2016).
- Shi, L. *et al. Nat. Commun.* **7**, 12065 (2016).
- Gordon, D. *et al. Science* **352**, aae0344 (2016).
- Berlin, K. *et al. Nat. Biotechnol.* **33**, 623–630 (2015).
- Chin, C.S. *et al. Nat. Methods* **13**, 1050–1054 (2016).
- Koch, P., Platzer, M. & Downie, B.R. *Nucleic Acids Res.* **42**, e80 (2014).
- Koren, S., Walenz, B.P., Berlin, K., Miller, J.R. & Phillippy, A.M. *Genome Res.* **27**, 722–736 (2017).
- Chaisson, M.J. & Tesler, G. *BMC Bioinformatics* **13**, 238 (2012).
- Myers, E.W. *Algorithmica* **1**, 251–266 (1986).
- Myers, G. *Algorithms in Bioinformatics*, 52–67 (2014).
- Li, H. Preprint at <https://arxiv.org/abs/1303.3997> (2013).
- Langmead, B. & Salzberg, S.L. *Nat. Methods* **9**, 357–359 (2012).

ONLINE METHODS

Indexing and matching of reads. The finding of potential matches between reads is based on the matching of k -mers (substrings with the length of k) in the reads. A read r of length L has a total of $L - k + 1$ k -mers. We first indexed the reads using a hash table with the k -mers as key. We considered the overlapping k -mers between the blocks of reads. We broke each read into multiple blocks; each block had a length B , which was usually 1,000 to 2,000 bp. The values in the hash table are the positions of k -mers in the blocks of reads.

To search for the matched reads, we scanned the k -mers in blocks of reads and found the matches in the hash table. We broke the reads into blocks of the same length B . To reduce the computer time, we sampled the k -mers in each search block. We used a sliding window with the length of sl along each block. Thus, the number of searched k -mers was approximately $1/sl$ of the number of total k -mers from the reads. A typical value of sl is 10. A searching block is matched to an indexing block if the number of their overlapping k -mers is greater than a predefined threshold m . The two reads are considered matched if at least a pair of blocks is matched between them.

Given two read blocks of length B , the number of k -mers sampled from the search block is $(B/sl - 1)$. With O as the overlapping length of a pair of matched blocks, $O \leq B$, the expected number of matched k -mers in O is¹¹

$$E[M_{\text{match}}] = (P_{\text{match}} + P_{\text{random}} - P_{\text{match}}P_{\text{random}}) \left(\frac{O}{sl} - 1 \right) + P_{\text{random}} \left(\frac{B - O}{sl} - 1 \right) \quad (1)$$

where P_{random} is the probability of the presence of a random k -mer, and P_{match} is the probability that two k -mers are matched. Because the block length B is fixed, for a given error rate and no repetitive sequence, the number of matched k -mers between two blocks grows with the overlapping length O . For a highly matched block pair, $P_{\text{match}} \gg P_{\text{random}}$, the expected number can be roughly estimated as:

$$E[M_{\text{match}}] = P_{\text{match}} \frac{O}{sl} \quad (2)$$

Filtering false-matched reads using the distance difference factor score. We developed a new pseudolinear scoring algorithm to filter the excessive, noninformative matched reads. Our scoring algorithm has two steps. The first step is the mutual scoring. For each matched read pair, we first randomly select a matched block pair and mark it. Then, we score the matched k -mer pairs in this matched block pair. Designating p_i, p_j as the positions of i -th and j -th k -mer in one block, and p'_i, p'_j as the positions of i -th and j -th k -mer in another block of a matched pair, we defined the distance difference factor (DDF _{i,j}) between i -th and j -th k -mer as

$$\text{DDF}_{i,j} = \left| 1 - \frac{p_i - p_j}{p'_i - p'_j} \right| \quad (3)$$

If $\text{DDF}_{i,j} < \varepsilon$, which indicates that both k -mers are supporting each other, we increase the scores of both k -mers by 1. The ε is set to 0.3; by calculating the DDF between all the possible pairs

of k -mers, we obtained scores for all the overlapping k -mers of matched blocks. We only used the nonrepetitive k -mer pairs in our scoring. If a k -mer was matched more than once, it was excluded from scoring. If the score of a k -mer with the highest score was significant (greater than the threshold), we set it as the seed position for future alignment. If there were multiple k -mers with the same score, we randomly selected one as the seed.

The second step was the extension scoring step. To increase the reliability of the seed and reduce the computation of the whole scoring process, we extended the scoring process from the selected block pair to its neighbor matched block pairs after a seed k -mer was obtained. For each overlapping k -mer in the neighbor block pair, we calculated the DDF between the k -mer and the seed k -mer in the original block pair. If $\text{DDF} < \varepsilon$, we increased the score of the seed k -mer by 1. If 80% of the DDF values of the overlapping k -mers in a neighbor block pair satisfied $\text{DDF} < \varepsilon$, we marked the block and did not score the k -mers in this block pair. If there were still unmarked matched block pairs after one loop of the mutual and extension scoring processes, we continued the scoring process on those block pairs. The mutual scoring is conducted in $O(N^2)$ time, and the extension scoring is conducted in $O(N)$ time, where N is the number of k -mer matches. Because the number of k -mers in mutual scoring is small, the overall scoring process can be performed in pseudolinear time.

Pairwise alignment of single-molecule sequencing reads. For pairwise alignment, we set the block length to 2,000 bp. After scoring the matched k -mers between two SMS reads, we sorted the k -mers based on their scores. Then, we used the top-ranked k -mers as seeds to perform the local alignment of the two reads. If the overlapped length between the two SMS reads was longer than 2,000 bp, and the mismatch rate of the overlapped sequence was less than twice that of the SMS read error rate, we considered it a match and output the alignment results. All the detailed parameters are described in **Supplementary Note 9**.

Aligning single-molecule sequencing reads to a reference genome. The procedure of aligning SMS reads to a reference genome is similar to that of pairwise alignment. We indexed the reference genome sequence and searched the reads from the index table. We first broke the reference genome into blocks with length B and indexed the k -mers in each block. Then, we broke the reads into blocks of the same length B and sampled the k -mers with a search in the index table. The matched k -mers between a read and the reference genome were also scored. The top-ranked k -mers were used as seeds to perform further local alignment.

To obtain high sensitivity of the alignment of the SMS reads to a reference genome, and to keep the computational cost low, we used a two-step approach. In the first step, we used the block length B of 1,000 bp and the k -mer sampling step length sl of 20 to align reads to the reference genome. Because some SMS reads have less matching k -mers, or the distribution of their matched k -mers is uneven, these SMS reads cannot find a matching position in the first step. In the second step, we doubled the block length B to 2,000 bp and halved the k -mer sample step length sl to 10; and we realigned the unmatched reads. We found the matches for a considerable number of reads in the first step and for most of the reads after the second step. Because the computational cost for the second step is higher than that for the first step, our two-step

approach allowed us to reduce the computational cost while maintaining high sensitivity. All the detailed parameters are described in **Supplementary Note 10**.

If there is a large structure variant in a read, the local alignment may be interrupted at the structure variant, which leads to unmapped tails of reads. If unmapped tail of a read was longer than 2,000 bp, we employed another step to find the alignment of this soft-clipped tail. For each read having soft-clipped tail, we selected its three longest matches in the reference genome. For each reference match of the read, if there were unmatched blocks close to the match, we examined whether the clipped tail could be matched to those unmatched blocks in reference genome. We scored the k-mer pairs between those unmatched blocks and clipped tail. If the score of top ranked k-mers was higher than the threshold, we performed the local alignment again to confirm the alignment.

Correcting single-molecule sequencing reads. Generally, there are two steps in correcting SMS reads. The first step is pairwise overlapping between SMS reads. The second step is constructing the correct read from the consensus of its related alignments. In MECAT, we adopted several approaches to improve the efficiency and accuracy of the consensus process. For the first step, we used MECAT aligner without local alignment for initial pairwise overlapping. The output of overlapping was written into multiple files. Each file included the matching information of 200,000 reads. Then, for each read template, we sorted its matches in order of descending DDF scores. We performed local alignments between the template read and matched reads starting from the highest DDF score. To eliminate the effects of chimeric reads and repeat subsequences, we filtered the alignment if its overlapped subsequence was less than 90% of the length of shorter read in the pair. The local alignment process was stopped once we collected 100 overlaps or had aligned all matched reads. Since the DDF score was a coarse estimate of the overlap length, by performing local alignment between read template and high-scored matches only, we were able to collect enough alignments for correction as soon as possible while avoiding the computing of noninformative repetitive overlaps. This significantly accelerated the error correction.

In the second step, to further improve the consensus precision while maintaining high efficiency, we developed a new adaptive SMS read error correction method by combining the principles from both DAGCon and FalconSense. We summarized the pairwise alignments to construct a consensus table with the counts of matches, insertions, and deletions. Trivial regions with consistent matches were designated as $\text{match_count}/(\text{match_count}+\text{deletion_count})>0.8$ and no significant insertion occurring ($\text{insertion_count}<6$); consistent deletions were designated as: $\text{deletion_count}/(\text{match_count}+\text{deletion_count})>0.8$ and no significant insertion occurring ($\text{insertion_count}<6$). Thus, we were able to determine the consensus base according to the count. For complicated regions with insertion ($\text{insertion_count}\geq 6$), we constructed a local POG and solved the consensus using dynamic programming. Because the complicated regions are generally fewer than ten bases, consensus sequences can be found quickly from the small POG. The details of this algorithm are described in **Supplementary Note 11** and **Supplementary Figure 2**.

In the second step of read correction, performing the local alignments between the template and matched reads requires random access to stored reads. DAGCon and FalconSense store the reads on the hard drive, which does not support random access. The slow loading process of the reads in DAGCon and FalconSense led to only a 20% CPU usage. To accelerate the correction process, we loaded all the reads into memory, which supports random access. We also encoded each base using 2 bits to reduce memory usage. Thus, the memory occupation of MECAT is approximately 1/4 of the total read size. Loading reads to memory renders the CPU usage of MECAT over 96%.

De novo assembly using single-molecule sequencing reads.

There are three steps in genome assembly using SMS reads: overlapping the SMS reads to the selected template reads, correcting the selected reads, and constructing the contigs using corrected reads. We developed two new pipelines for assembling SMS reads by integrating our new alignment and error correction method with Canu (v1.0). In the first step of the MECAT pipeline, for each read longer than 3,000 bp, we performed a pairwise alignment against other reads and selected 100 matched reads with top-matched scores. During the overlapping of the SMS reads, we did not perform local alignment. We selected the top-mapped reads using the DDF scores and used the mapping information for the error correction step. In the second step of MECAT, we corrected all template reads (>3,000 bp) using their matched reads. Finally, we performed a pairwise alignment of corrected reads using the alignment tool in MECAT; then, we fed the results of the alignment into the 'Unitig Construction' module of Canu (v1.0) to construct the unitigs. Alternatively, in pipeline MECAT-CA, the corrected reads were fed directly into Canu, which used the overlapInCorefor pairwise alignment.

Evaluation. We evaluated the MECAT tool using both simulated and raw SMS reads from model organisms. We compared our alignment tool with the existing tools for pairwise alignment, including MHAP and DALIGNER, as well as with the tools for reference genome alignment, including BLASR and BWA-mem (**Supplementary Notes 1–4**). We compared our error correction tool with those available in Canu (v1.3) and FALCON (v0.40). We also systematically evaluated the assembly tools available in MECAT by comparing them with Canu (v1.3) and FALCON (v0.40). The details of these comparisons are reported in the **Supplementary Notes 5–8**.

Data availability statement. The raw sequencing data of the Han-1 Chinese human genome are available from GenBank (SRX1424851). The assembly files of the Han-1 Chinese human genome are available from GenBank (GCA_001856745.1). All source codes for MECAT and the analyses presented here are available from <https://github.com/xiaochuanle/MECAT>. The software and data used for this manuscript (including supplementary files and scripts) are available from <http://sysbio.sysu.edu.cn/MECAT>.

A **Life Sciences Reporting Summary** is available.

Life Sciences Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form is intended for publication with all accepted life science papers and provides structure for consistency and transparency in reporting. Every life science submission will use this form; some list items might not apply to an individual manuscript, but all fields must be completed for clarity.

For further information on the points included in this form, see [Reporting Life Sciences Research](#). For further information on Nature Research policies, including our [data availability policy](#), see [Authors & Referees](#) and the [Editorial Policy Checklist](#).

▶ Experimental design

1. Sample size

Describe how sample size was determined.

N/A

2. Data exclusions

Describe any data exclusions.

N/A

3. Replication

Describe whether the experimental findings were reliably reproduced.

N/A

4. Randomization

Describe how samples/organisms/participants were allocated into experimental groups.

N/A

5. Blinding

Describe whether the investigators were blinded to group allocation during data collection and/or analysis.

N/A

Note: all studies involving animals and/or human research participants must disclose whether blinding and randomization were used.

6. Statistical parameters

For all figures and tables that use statistical methods, confirm that the following items are present in relevant figure legends (or in the Methods section if additional space is needed).

n/a Confirmed

- The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement (animals, litters, cultures, etc.)
- A description of how samples were collected, noting whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
- A statement indicating how many times each experiment was replicated
- The statistical test(s) used and whether they are one- or two-sided (note: only common tests should be described solely by name; more complex techniques should be described in the Methods section)
- A description of any assumptions or corrections, such as an adjustment for multiple comparisons
- The test results (e.g. P values) given as exact values whenever possible and with confidence intervals noted
- A clear description of statistics including central tendency (e.g. median, mean) and variation (e.g. standard deviation, interquartile range)
- Clearly defined error bars

See the web collection on [statistics for biologists](#) for further resources and guidance.

▶ Software

Policy information about [availability of computer code](#)

7. Software

Describe the software used to analyze the data in this

All source codes for MECAT, and the analyses presented here, are available from

study.

<https://github.com/xiaochuanle/MECAT>. The software and data used for this manuscript (including supplementary files and scripts) are available from <http://sysbio.sysu.edu.cn/software/MECAT>.

For manuscripts utilizing custom algorithms or software that are central to the paper but not yet described in the published literature, software must be made available to editors and reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). *Nature Methods* [guidance for providing algorithms and software for publication](#) provides further information on this topic.

► Materials and reagents

Policy information about [availability of materials](#)

8. Materials availability

Indicate whether there are restrictions on availability of unique materials or if these materials are only available for distribution by a for-profit company.

N/A

9. Antibodies

Describe the antibodies used and how they were validated for use in the system under study (i.e. assay and species).

N/A

10. Eukaryotic cell lines

a. State the source of each eukaryotic cell line used.

N/A

b. Describe the method of cell line authentication used.

N/A

c. Report whether the cell lines were tested for mycoplasma contamination.

N/A

d. If any of the cell lines used are listed in the database of commonly misidentified cell lines maintained by [ICLAC](#), provide a scientific rationale for their use.

N/A

► Animals and human research participants

Policy information about [studies involving animals](#); when reporting animal research, follow the [ARRIVE guidelines](#)

11. Description of research animals

Provide details on animals and/or animal-derived materials used in the study.

N/A

Policy information about [studies involving human research participants](#)

12. Description of human research participants

Describe the covariate-relevant population characteristics of the human research participants.

N/A