Bi-level error correction for PacBio long reads

Yuansheng Liu, Chaowang Lan, Michael Blumenstein, and Jinyan Li

Abstract—The latest sequencing technologies such as the Pacific Biosciences (PacBio) and Oxford Nanopore machines can generate long reads at the length of thousands of nucleic bases which is much longer than the reads at the length of hundreds generated by Illumina machines. However, these long reads are prone to much higher error rates, for example 15%, making downstream analysis and applications very difficult. Error correction is a process to improve the quality of sequencing data. Hybrid correction strategies have been recently proposed to combine Illumina reads of low error rates to fix sequencing errors in the noisy long reads with good performance. In this paper, we propose a new method named Bicolor, a bi-level framework of hybrid error correction for further improving the quality of PacBio long reads. At the first level, our method uses a de Bruijn graph-based error correction idea to search paths in pairs of solid k-mers iteratively with an increasing length of k-mer. At the second level, we combine the processed results under different parameters from the first level. In particular, a multiple sequence alignment algorithm is used to align those similar long reads, followed by a voting algorithm which determines the final base at each position of the reads. We compare the superior performance of Bicolor with three state-of-the-art methods on three real data sets. Results demonstrate that Bicolor always achieves the highest identity ratio. Bicolor also achieves a higher alignment ratio ($> 1.3\%$) and a higher number of aligned reads than the current methods on two data sets. On the third data set, our method is closely competitive to the current methods in terms of number of aligned reads and genome coverage. The C++ source codes of our algorithm are freely available at https://github.com/yuansliu/Bicolor.

Index Terms—error correction, PacBio long reads, de Bruijn graph, multiple sequence alignment

1 Introduction

The second generation sequencing technologies, which are high-throughput with low costs and high quality, have been employed successively in many applications, including resequencing, *de novo* sequencing, transcriptome profiling and metagenomics [1], [2], [3]. However, it produces relatively short reads—the median length of the reads produced by Illumina is 100 bp. Short reads largely decrease the continuity and provide less information to process the repetitive subsequences [4], thus having difficulty in assembling. Newer next-generation sequencing (NGS) technologies [5], for example the Pacific Biosciences and Oxford Nanopore platforms, can produce long reads at the length up to $50,000$ bp. The long reads offer much more information than the short reads to resolve the issue of complex repetitions. In Pacific Biosciences Real-time Sequencer, the higher overall error rate of earlier chemistries, which is approximately two orders of magnitude than that of Illumina platforms [6], result in the long reads having much higher error rates (at least 15%) [7]. The drawback of extremely high error rates poses a challenge for downstream analysis and applications [6], [8], [9], [10].

Although many algorithms have been developed for correcting short reads [6], [11], [10], these algorithms are not directly applicable for correcting long reads. This is because the long reads are dominated by insertion and deletion (indels) errors—indels are about 15 times more common than substitution, while the major error type of short reads is substitution. Recently, several algorithms have been proposed for long read error correction. These algorithms can be classified into two categories according to whether or not short reads are used. The first category is a self-correction approach, which only uses noisy long reads, including the methods HGAP [12], Canu [13], and LoRMA [14]. There are many limitations in the self-correction approach, such as the required high coverage and the substantial computational cost [15]. Therefore, the second category called hybrid-correction have been developed to enhance the performance of long reads error correction.

The hybrid-correction approach makes use of the short reads to correct the errors in the long reads. As short reads have lower error rate (about 1%) than long reads [10], the short reads provides a good template for the long reads correction. The hybrid-correction approach has two main ideas. The first one is that it builds mappings between the short reads and the long reads, then corrects long reads through the mapping. For example, pacBioToCA [16] uses the mapping information to select the overlaps that are converted into a tiling of short read sequences along each long read. A new consensus sequence is then generated for each long read via a multiple-alignment of the tiled short read sequences [17]. LSC [18] employs a homopolymer compression (HC) transformation prior to the mapping. Then, it discovers four types of correction points: HC points, mismatches, deletions, and insertions. These points are replaced by their short read consensus sequences. The method proovread [7] computes the consensus by using the mapping information and a vote strategy. The novelty of proovread is the iterative correction step, which consists of three pre-correction and one finishing cycles. CoLoRMap [15] builds a weighted alignment graph based on the mapping information. Then, a classical shortest path algorithm is applied to construct the corrected region with the minimum edit score. For some regions of a long read

- *Y. Liu, C. Lan and J. Li are with the Advanced Analytics Institute, University of Technology Sydney, Broadway, NSW 2007, Australia.E-mail: yyuanshengliu@gmail.com, Jinyan.Li@uts.edu.au. Correspondence should be addressed to J. Li.*
- *M. Blumenstein is with the School of Software, Faculty of Engineering and Information Technology, University of Technology Sydney, Broadway, NSW 2007, Australia.*

that are not covered by the short reads, One-End-Anchors (OEA) are used to expand the corrected regions.

However, these methods map short reads individually and do not exploit the context in which the short read occurs [19]. Other methods, such as LoRDEC [20] and Jabba [19], construct a de Bruijn graph (DBG) from the short reads, then use sequence alignment algorithms to align the long reads to the DBG. LoRDEC [20] aligns the long reads to the DBG by finding an optimal path such as to minimize the edit distance between two solid k-mers of the long read. Jabba employs the seed-and-extend strategy to align the long read to the DBG. These methods have a common limitation that the quality of the long reads correction heavily depends on the length of k-mer. If a user sets a large k-mer, only a few DBGs can be mapped to the long reads. Thus, many wrong base pairs cannot be corrected. On the other hand, if the user sets a small k-mer, a lot of DBGs can be mapped to the long reads, making it difficult to opt the final result.

In this paper, we propose a new method named Bicolor to improve the quality of long reads. Our method has two levels of processing. At the first level, we set a strict condition for the selection of solid k-mers. The selection criteria overcomes the limitation that the length of k-mer affects on the quality of mapping the long reads to the DBG. Then the long reads are iteratively corrected by using several k-mers of different length. Therefore, we can obtain several pre-corrected long reads under different initial lengths of k-mer. At the second level, we utilize the multiple sequence alignment (MSA) algorithm to align these similar pre-corrected long reads [21], and then use a vote algorithm to get the final corrected long read. The key idea of our method is to combine the sets of pre-corrected long reads, derived by using k-mers of different lengths. Experiment results show that our method achieves better performance than the state-of-the-art error correction methods.

2 Methods

Our algorithm Bicolor is a bi-level framework for noisy long reads error correction. A schematic diagram of Bicolor is depicted in Fig. 1.

The first level consists of n iterative correctors each using a k-mer of different length. The iterative corrector iteratively corrects the noisy long read m times under its initial k-mer. The initial k-mer of this iterative corrector increases its size k in the subsequent iteration. Thus, we can obtain n pre-corrected long reads in the first level. Then these pre-corrected long reads are processed by MSA and a vote algorithm in the second level. The output of the second level is the final corrected long reads.

2.1 First level: long read pre-correction

Iterative correction is the core of the first level computation. Similar iterative approaches has been used for short reads assembly [22], [23], short reads correction [24], and self-correction [14]. LoRDEC [20] is modified to an iterative version (called iLoREDC) to perform the computation. There are three main steps in LoRDEC: (1) constructing a DBG using short reads; (2) determining solid/weak k-mers in long read; and (3) searching path in the DBG with minimal edit distance between two solid k-mers. The DBG is the core of most second-generation assemblers such as Velvet [25], Minia [26]. DBG connects short reads into a graph. Then a long read can align to the DBG by finding solid k-mers. Here, solid k-mers in long reads are preserved as correct substrings which are assumed to have no errors. We assume that errors only exist in weak k-mers. Therefore, weak k-mers can be corrected by searching paths between solid k-mers.

Let L be a noisy long read, an odd integer k be an initial length of k-mers and m be the number of iterations. The procedure of iterative correction by iLoRDEC is as follows:

- Step 1: Use the short reads to build a DBG, where an edge connects two nodes if their corresponding k-mers are overlapped by $(k-1)$ bases. These k-mers that occur less than s times within the short reads are filtered out.
- Step 2: Find solid k-mers in a long read L. Given all k-mers of a long read L, if both the i-th k-mer and the $(i+1)$-th k-mer of L are in the DBG, the i-th k-mer of L is a solid k-mer, otherwise it is a weak k-mer. One or more consecutive solid k-mers construct a solid region and one or more consecutive weak k-mers form a weak region. Specially, the weak regions located at the beginning and the end of the long reads are called the head region or the tail region, respectively.
- Step 3: Correct weak regions of L. Find a path between the solid regions of L in the DBG to correct the weak regions. If several paths are found, the path with minimal edit distance is selected as the corrected sequence.
- Step 4: Correct these head and tail regions by searching a path with minimal edit distance to these regions.
- Step 5: Use the Dijkstra algorithm to find the shortest path between the first and the last solid k-mers.
- Step 6: Update $m = m - 1$ and $k = k + 2$.
- Step 7: If $m > 0$, go to Step 1 and use the corrected sequence as the input. Otherwise, output this corrected sequence. The output of the corrected sequence is called pre-corrected long read.

Details of Steps 3, 4 and 5 can be seen in [20].

Several modifications are made by iLoRDEC in comparison with LoRDEC:

1) In Step 2, we strengthen the selection of solid k-mers. By LoRDEC, if the i-th k-mer of L is in the DBG, it is treated as solid. If we use a large k, the long read may not contain a solid k-mer. Thus, the error base-pairs in the long read would not be corrected. If we set a small k, the long read can have many solid k-mers. The long read may be over-corrected as the repeats of sequence and false positive of solid k-mers often exist in a long read. To overcome this issue, we select only the first k-mer as solid one iff two consequent k-mers of the long read exist in the DBG. This selection criteria can improve the reliability of solid k-mers.
2) iLoRDEC only performs one pass in Steps 3, 4 and 5, while LoRDEC performs two passes on two direc-

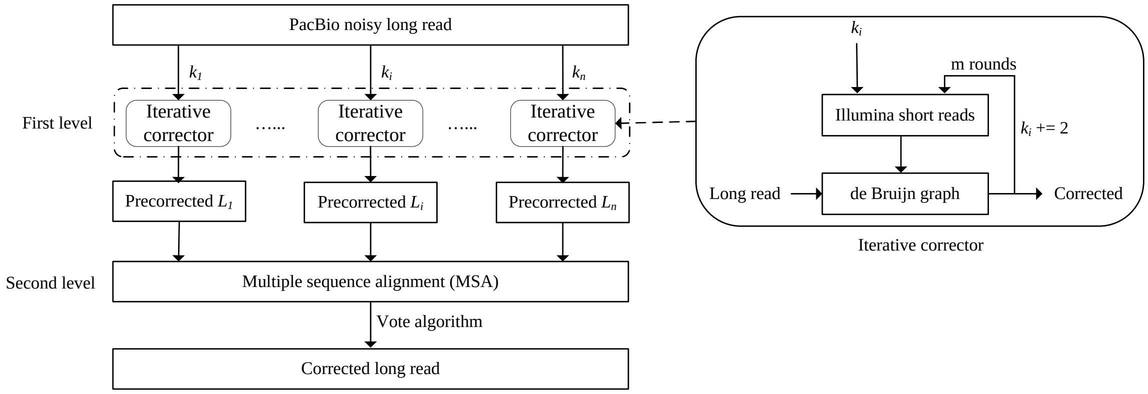


Fig. 1. Schematic diagram of our algorithm Bicolor.

tions. LoRDEC corrects the reverse complementary of the long read and outputs a corrected long read in the first pass. In the second pass, LoRDEC transforms the corrected long read to its reverse complementary sequence and corrects this sequence. The following two reasons motivate Salmela and Rivals [20] to perform two passes: (1) new solid k-mers are used as starting nodes in the next pass; (2) different region's ending leads to different paths. Actually, iLoRDEC is an iterative algorithm, new solid k-mers are used as both starting or ending nodes in the subsequent rounds of iteration. Therefore, we do not consider the reverse complementary of the long read.

3) We add Steps 6 and 7 to iterate different length k-mers with m rounds, each round k is increased by 2.

There are n iterative correctors in the first level. Each corrector iteratively corrects the long read by using different initial lengths of k-mer. Therefore, we can obtain n pre-corrected long reads at this level.

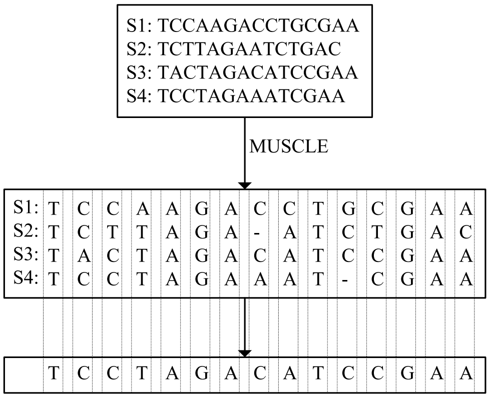
2.2 Second level: MSA-based correction

MSA has been widely used in the current molecular biology, such as inferring sequence homology [27], improving protein secondary structure prediction [28] and conducting phylogenetic analysis [29]. At the second level of our correction framework, MSA is used to align those pre-corrected long reads derived from the first level. The tool MUSCLE [30] is applied in our implementation. A simple vote algorithm is subsequently utilized to generate the final corrected sequence. This simple vote algorithm selects the most frequent bases as the final result at each position.

For illustration, an example with 4 sequences is depicted in Fig. 2, where the 4 sequences are 4 pre-corrected long reads. We use the MUSCLE to align these pre-corrected long reads. As the second base of S1, S2, and S4 is C and the second base of S3 is A, the most frequent base in the second position of these pre-corrected long read is C. Then, the second base of the final corrected read is C.

3 Results and Analysis

The correction results and some analysis are presented in this section. The performance of our proposed algorithm


S1: TCCAAGACCTGCGAA
S2: TCTTAGAATCTGAC
S3: TACTAGACATCCGAA
S4: TCCTAGAAATCGAA

S1:	T	C	C	A	A	G	A	C	C	T	G	C	G	A	A
S2:	T	C	T	T	A	G	A	-	A	T	C	T	G	A	C
S3:	T	A	C	T	A	G	A	C	A	T	C	C	G	A	A
S4:	T	C	C	T	A	G	A	A	A	T	-	C	G	A	A
	T	C	C	T	A	G	A	C	A	T	C	C	G	A	A

Fig. 2. An example to illustrate the second level correction.

Bicolor is benchmarked in comparison with three existing algorithms: LoRDEC [20], CoLoRMap+OEA [15], and CoLoRMap [15]. As reported in [15], [20], CoLoRMap and LoRDEC had achieved comparable performance when comparing with pacBioToCA, LSC and proovread. We did not compare our performance directly with pacBioToCA, LSC or proovread. All the experiments were conducted on a computing cluster running Red Hat Enterprise Linux 6.7 (64 bit) with 2 × 2.3 GHz Intel Xeon E5-2695 v3 (14 Cores) and 128 GB RAM.

3.1 Data sets

The algorithms are tested on three data sets: a bacterial genome from *Escherichia coli* (E. coli), two eukaryotic genomes from *Saccharomyces cerevisiae* (yeast) and *Drosophila melanogaster* (fruit fly). They are benchmark data sets used in [15]. More details of these data sets are shown in Tab. 1.

3.2 Comparison with LoRDEC, CoLoRMap and CoLoRMap+OEA

In the performance comparison of Bicolor with algorithms LoRDEC [20], CoLoRMap [15] and CoLoRMap+OEA [15], the default parameter settings were used (see Tab. 2). To measure the performance by the correction methods, we used BLASR [31] to align long reads to the reference genome. For each read, we store a single best alignment

TABLE 1
Details of the three benchmark data sets

		Bacteria	Yeast	Fruit fly
Organism	Name	Escherichia coli	Saccharomyces cerevisiae	Drosophila melanogaster
	Strain	K-12 substr. MG1655	S288C	ISO1
	Reference sequence	NC_00913	NC_0011{33-48} NC_001224	NT_0337{77-79}; NC_0043{53-54}; NC_0245{11-12}; NT_037436
	Genome size	4.6 Mbp	12.2 Mbp	137.6 Mbp
PacBio data[1]	Download from	DevNet[2]	DevNet[3]	Bergman Lab[4]
	Number of reads	33,360	261,964	901,530
	Max read length	14,494	30,164	13,885
	Avg read length	2,938	5,891	1,505
	Number of bases	98,015,299	1,543,321,663	1,357,180,677
Illumina data	Accession ID	ERR022075[5]	SRR567755	ERX645969[5]
	Number of reads	2,316,614	4,503,422	70,000,000
	Read length	100&102	101	101
	Number of bases	233,978,014	454,845,622	7,070,000,000

[1] All long reads whose length less than 100 bp were filtered out.
[2] https://github.com/PacificBiosciences/DevNet/wiki/E-coli-K12-MG1655-Hybrid-Assembly
[3] https://github.com/PacificBiosciences/DevNet/wiki/Saccharomyces-cerevisiae-W303-Assembly-Contigs
[4] http://bergmanlab.genetics.uga.edu/data/genomes/2057_PacBio.tgz
[5] Only a subset of the data was used.

under the options '-noSplitSubreads -bestn 1'. We then computed the following statistics as metrics:

- *Number of aligned reads*: the number of long reads that align to the reference genome.
- *Alignment ratio*: the ratio between the number of aligned bases and the total bases of long reads.
- *Identity ratio*: the ratio between the number of matched bases and the length of the aligned region in the reference genome.
- *Genome coverage*: the proportion of the genome aligned regions by long reads.

The number of aligned reads measures the throughput of the correction algorithm. A bigger number of long reads aligned to the reference genome stands for that more noisy long reads are corrected. The alignment ratio and the identity ratio stands for the quantity and quality of the aligned bases respectively. They together measure the accuracy of correction. Genome coverage defines the extent to which the reference genome is covered by the corrected reads. This evaluation approach has been widely adopted by the state-of-the-art methods [20], [15], [18], [14].

TABLE 2
Default parameters of the three existing methods

Data set / Method	E. coli	Yeast	Fly
LoRDEC [20]	-k 19 -s 3 -e 0.4 -b 200 -t 5		
CoLoRMap [15]	BWA-MEM: -a Y -A 5 -B 11 -O 2,1 -E 4,3 -k 8 -W 16 -w 40 -r 1 -D 0 -y 20 -L 30,30 -T 2.5; Mina: -kmer-size 43 -abundance-n 1		
Bicolor	$k_1 = 13, k_2 = 15, k_3 = 17, k_4 = 19, s = 3, e = 0.4, b = 200, t = 5$		
	$m_i = 3$	$m_i = 4$	$m_i = 4$

The comparison results are shown in Tab. 3. On the data set E. coli, all the methods can achieve a close performance in terms of identity ratio (above 99%), where our method is the highest. The number of reads aligned back to the reference genome by Bicolor is at least 471 much more than the other methods. Compared with LoRDEC and CoLoRMap, our alignment ratio is improved by 3.2% and 1.7% respectively. While the alignment ratios of LoRDEC and CoLoRMap even less than that of the original noisy long reads without any correction.

On the data set yeast, the corrected reads by Bicolor can align $246,122$ of them back to the reference genome. This number exceeds the other methods by at least $4,548$. The alignment ratio achieved by Bicolor is 83.442%, which is 2.7% and 1.3% higher than LoRDEC's alignment ratio 80.672% and CoLoRMap's alignment ratio 82.072. Bicolor also achieved the highest identity ratio 97.969%, which is higher than LoRDEC's identity ratio 97.810% and 1.4% higher than CoLoRMap's identify 96.515%.

On the third data set fruit fly, the corrected long reads by Bicolor align less number of reads back to the reference genome than that of LoRDEC, while Bicolor can achieve a higher alignment ratio and identity ratio. Bicolor has 4413 more number of aligned reads compared with CoLoRMap, and can also achieve a higher identity ratio. We note that CoLoRMap can have a 2.1% higher alignment ratio than Bicolor (37.544% identity ratio). It can be seen that this data set has many erroneous bases, because there are only $313,989$ among $901,530$ reads can align to the reference genome and the raw data has a relative low alignment ratio (only 37.079%). This has lead to solid k-mers in the long reads extremely unreliable for correction. Furthermore, the searched paths in the DBG are far from the expected ones. On the other hand, CoLoRMap can align short reads to long reads and dose not rely on solid k-mers. Even more reads are aligned to the reference genome after correcting by Bicolor, it achieves lower alignment ratio than that of CoLoRMap. It is worth of noting that Bicolor achieves the highest identity ratio.

All the methods have very close performance under the typical adopted genome coverage (Tab. 4). It can be still understood that CoLoRMap performed best. On the data set

TABLE 3
Alignment performance by different methods on three data sets

Data set	Method	No. of aligned reads	Alignment ratio (%)	Identity ratio(%)
E. coli	Original*	31,071	88.429	94.799
	LoRDEC	30,837	86.942	99.444
	CoLoRMap	31,018	88.401	99.006
	CoLoRMap+OEA	30,939	87.996	99.119
	Bicolor	**31,489**	**90.178**	**99.467**
Yeast	Original	239,232	80.449	93.079
	LoRDEC	240,413	80.672	97.810
	CoLoRMap	241,574	82.072	96.393
	CoLoRMap+OEA	241,571	82.070	96.515
	Bicolor	**246,122**	**83.442**	**97.969**
Fly	Original	313,989	37.079	94.600
	LoRDEC	**342,800**	37.364	97.091
	CoLoRMap	337,799	**39.630**	97.901
	CoLoRMap+OEA	337,799	39.629	97.956
	Bicolor	342,212	37.544	**98.041**

*Original is the alignment statistics without being corrected by algorithm.

E. coli, all the methods can achieve the 100% coverage. On the data set yeast, CoLoRMap performs slightly better than LoRDEC and Bicolor. Specifically, the coverage by Bicolor and LoRDEC are only 0.012% and 0.03% less than that of CoLoRMap. Also, Bicolor obtains the lowest coverage (i.e., 93.915%) on the data set fruit fly, which is 0.063% and 0.82% less than that of LoRDEC and CoLoRMap, whose coverages are 93.978% and 94.735%, respectively.

TABLE 4
Comparison on genome coverage

Method	Data sets		
	E. coli	Yeast	Fly
Original	100	99.793	93.657
LoRDEC	100	99.822	93.978
CoLoRMap	100	99.852	94.735
CoLoRMap+OEA	100	99.852	94.729
Bicolor	100	99.840	93.915

3.3 Performance improvement from LoRDEC to iLoRDEC

In the first level, we polish LoRDEC to an iterative version. We compare the performance of iLoRDEC and LoRDEC in this subsection. In order to compare with LoRDEC, we perform some experiments on the data set E. coli with some different parameters. The alignment statistics of long reads corrected by iLoRDEC with different initial k-mers and several different numbers of iterative rounds are shown in Tab. 5. In [20], Salmela and Rivals have claimed that LoRDEC achieve best result (see second row of Tab. 3) under default parameters. Comparing with the best result of LoRDEC, we find that iLoRDEC performs better than LoRDEC under six group parameters. It is worth noting that iLoRDEC always achieves higher alignment ratio. These verify the effectiveness of iLoRDEC.

TABLE 5
Alignment statistics of E. coli data corrected by iLoRDEC under different parameters

Parameters		No. of aligned reads	Alignment ratio (%)	Identity ratio(%)
$k = 13$	$m = 1$	30,679	**87.409**	93.296
	$m = 2$	**30,987**	**88.382**	95.432
	$m = 3$	**31,210**	**89.189**	97.190
	$m = 4$	**31,054**	**88.308**	97.906
	$m = 5$	30,816	**87.074**	98.380
$k = 15$	$m = 1$	**31,257**	**89.366**	97.456
	$m = 2$	30,782	**87.071**	99.186
	$m = 3$	30,830	**87.246**	99.377
	$m = 4$	**30,868**	**87.464**	**99.448**
	$m = 5$	**30,942**	**87.760**	**99.484**
$k = 17$	$m = 1$	30,810	**87.002**	99.329
	$m = 2$	**30,847**	**87.066**	**99.483**
	$m = 3$	**30,872**	**87.219**	**99.517**
	$m = 4$	**30,888**	**87.305**	**99.529**
	$m = 5$	**30,898**	**87.356**	**99.536**
$k = 19$	$m = 1$	**30,911**	**87.368**	99.264
	$m = 2$	**30,875**	**87.115**	99.381
	$m = 3$	**30,877**	**87.146**	99.401
	$m = 4$	**30,875**	**87.175**	99.411
	$m = 5$	**30,876**	**87.178**	99.417

Bold indicates the corresponding value better than that of LoRDEC.

3.4 Effectiveness of MSA-based correction

After correcting by iLoRDEC, we get n pre-corrected long reads. Then, MUSCLE is used to align these similar long reads in the second level. In order to verify the effectiveness of MSA-based correction, we combine the results, which are corrected by iLoRDEC with four different initial k-mers (i.e., $n = 4$) and five different numbers of iterative rounds on the data set E. coli, to obtain final corrected long reads. The alignment statistics of final corrected results are shown in Tab. 6. Comparing the alignment statistics in Tabs. 6 and 5, we can see that the results after correcting by MSA are much better than that of iLoRDEC regarding number of aligned reads and alignment ratio. In addition, the identity ratio is very close to the highest identity ratio in Tab. 5. The results imply that using several sets of pre-corrected long reads to get the final corrected long reads can enhance the performance.

3.5 Parameters setting for the optimal time costs

The initial length of k-mer, number of iterative corrector n and rounds number m at the first level are the most important parameters in our method Bicolor. Other four parameters, i.e., the threshold for solid k-mers, the maximum error rate and branching limit and the number of target k-mer, inherited from LoRDEC, are set as the default values by LoRDEC (see Tab. 2). If k_i is large, many long reads can not be corrected because they may not contain any solid k-mers. We suggest that the initial length of k-mer used by iLoRDEC should be smaller than the default value used by LoRDEC. But, a smaller k_i will result in a DBG of higher complexity, causing the running time of iLoRDEC much longer. Following the instructions of LoRDEC, the initial length of k-mer is suggested to be within the set $\{13, 15, 17, 19\}$ for bacterial

TABLE 6
Alignment statistics of long reads corrected by MSA

Parameters	No. of aligned reads	Alignment ratio (%)	Identity ratio(%)
$k_1 = 13, m_1 = 1$ $k_2 = 15, m_2 = 1$ $k_3 = 17, m_3 = 1$ $k_4 = 19, m_4 = 1$	$31,401$	90.006	98.526
$k_1 = 13, m_1 = 2$ $k_2 = 15, m_2 = 2$ $k_3 = 17, m_3 = 2$ $k_4 = 19, m_4 = 2$	$31,551$	90.465	99.360
$k_1 = 13, m_1 = 3$ $k_2 = 15, m_2 = 3$ $k_3 = 17, m_3 = 3$ $k_4 = 19, m_4 = 3$	$31,489$	90.178	99.467
$k_1 = 13, m_1 = 4$ $k_2 = 15, m_2 = 4$ $k_3 = 17, m_3 = 4$ $k_4 = 19, m_4 = 4$	$31,344$	89.515	99.503
$k_1 = 13, m_1 = 5$ $k_2 = 15, m_2 = 5$ $k_3 = 17, m_3 = 5$ $k_4 = 19, m_4 = 5$	$31,193$	88.685	99.518

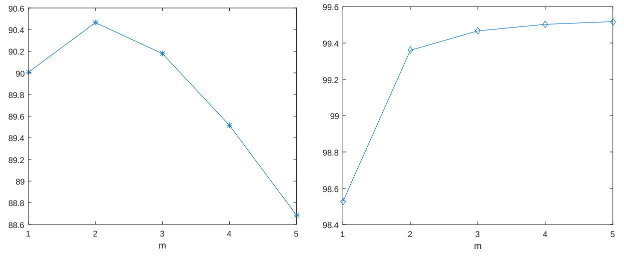
and eukaryotic species of small genomes. For large-genome species, we suggest $k_i \in \{13, 15, 17, 19, 21\}$. It has been observed that Bicolor's performance degrades as iLoRDEC's when $n = 1$. Considering both the vote algorithm and running time, we suggest $n \geq 3$.

Selection of a good number of iterative rounds is tricky. Fig. 3 shows a trend of the alignment ratios and identity ratios under four different initial k-mers and five different numbers of iterative rounds (Tab. 6). From this figure, we can see that the alignment ratio can reach to the highest level when the number of iterative rounds becomes 2. In addition, as the number of iterative rounds increases from 2 to 5, the alignment ratio is decreased. However, when the number of iterative rounds is smaller than 4, the alignment ratio is still relatively high (more than 90%). Thus, if the best iterative round is less than 4, we can obtain a good alignment ratio. This figure also indicates that the identity ratio is proportional to the number of iterative rounds. This is because the higher the number of iterative rounds is, the more errors are corrected. However, the identity ratio is not significantly increased after the number of iterative is set larger than 3. So we can obtain a better identity ratio if the iterative round is set larger than 2. Also, the running time can be significantly longer when the number of iterative rounds is increased. Therefore, we suggest that the number of iterative rounds should be less than 5. It is expected that the correction result should have relatively high alignment ratio, high identify ratio, and low time consumption. In this work, we suggest the iterative round as 3 or 4.

At the second level, we set the fastest option '-maxiters 1 -diags' of MUSCLE in our experiments since time-consuming MSA is of high complexity.

3.6 Running time comparison

To compare the running time of these methods, we use the Linux/Unix time command to record the real time. In our experiments, all cores are used to run the programs. The running time of these methods is reported in Tab. 7. LoRDEC is the fastest method. As CoLoRMap is mapping-based method, thus it is slower than LoRDEC. Especially, the procedure of OEA is very time-consuming. Bicolor contains two stages of computation. The first stage has a number of iLoRDEC. It's expected that the running time is many times longer than LoRDEC, even though we did some improvements. Another reason is that the complexity of MSA is very high. We used the fastest option of MUSCLE, but it still spent much time. Bicolor run faster than CoLoRMap+OEA only.



(a) (b)

Fig. 3. Alignment ratio (a) and identity ratio (b) under different iterative rounds.

TABLE 7
Comparison of running time (minutes) on different data sets

Method	Data sets		
	E. coli	Yeast	Fly
LoRDEC	5	48	82
CoLoRMap	21	131	400
CoLoRMap+OEA	93	2866	8399
Bicolor	90	1462	1138

4 Conclusion

This paper has introduced a bi-level framework for the error correction of PacBio long reads. At the first level, it utilizes k-mers of different lengths and an iterative algorithm to determine multiple sets of preliminarily corrected reads. Then our method combines these preliminary results by MSA-based correction at the second level. The performance evaluation on three benchmark data sets has demonstrated that our proposed method can achieve the highest identity ratio in comparison with three state-of-the-art algorithms. The performance on the alignment ratio has been improved on the data sets E. coli and yeast. Our method also has some drawbacks. First, there is a little genome coverage lost on the data sets yeast and fruit fly. Second, the running time is longer than the other methods except the OEA method. Our future work will focus on these areas for speed improvement.

Acknowledgements

This work was partly supported by the Australia Research Council Discovery Project DP130102124.

REFERENCES

[1] 1000 Genomes Project Consortium and others, "A map of human genome variation from population-scale sequencing," *Nature*, vol. 467, no. 7319, pp. 1061–1073, 2010.

[2] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt *et al.*, "The sequence of the human genome," *science*, vol. 291, no. 5507, pp. 1304–1351, 2001.

[3] J. Pell, A. Hintze, R. Canino-Koning, A. Howe, J. M. Tiedje, and C. T. Brown, "Scaling metagenome sequence assembly with probabilistic de Bruijn graphs," *Proceedings of the National Academy of Sciences*, vol. 109, no. 33, pp. 13 272–13 277, 2012.

[4] C. Kingsford, M. C. Schatz, and M. Pop, "Assembly complexity of prokaryotic genomes using short reads," *BMC bioinformatics*, vol. 11, no. 1, p. 21, 2010.

[5] S. Goodwin, J. D. McPherson, and W. R. McCombie, "Coming of age: ten years of next-generation sequencing technologies," *Nature Reviews Genetics*, vol. 17, no. 6, pp. 333–351, 2016.

[6] D. Laehnemann, A. Borkhardt, and A. C. McHardy, "Denoising DNA deep sequencing data—high-throughput sequencing errors and their correction," *Briefings in bioinformatics*, vol. 17, no. 1, pp. 154–179, 2016.

[7] T. Hackl, R. Hedrich, J. Schultz, and F. Förster, "proovread: large-scale high-accuracy PacBio correction through iterative short read consensus," *Bioinformatics*, vol. 30, no. 21, pp. 3004–3011, 2014.

[8] M. G. Ross, C. Russ, M. Costello, A. Hollinger, N. J. Lennon, R. Hegarty, C. Nusbaum, and D. B. Jaffe, "Characterizing and measuring bias in sequence data," *Genome biology*, vol. 14, no. 5, p. R51, 2013.

[9] N. J. Loman, R. V. Misra, T. J. Dallman, C. Constantinidou, S. E. Gharbia, J. Wain, and M. J. Pallen, "Performance comparison of benchtop high-throughput sequencing platforms," *Nature biotechnology*, vol. 30, no. 5, pp. 434–439, 2012.

[10] A. S. Alic, D. Ruzafa, J. Dopazo, and I. Blanquer, "Objective review of de novo stand-alone error correction methods for NGS data," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 6, no. 2, pp. 111–146, 2016.

[11] X. Yang, S. P. Chockalingam, and S. Aluru, "A survey of error-correction methods for next-generation sequencing," *Briefings in bioinformatics*, vol. 14, no. 1, pp. 56–66, 2013.

[12] C.-S. Chin, D. H. Alexander, P. Marks, A. A. Klammer, J. Drake, C. Heiner, A. Clum, A. Copeland, J. Huddleston, E. E. Eichler *et al.*, "Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data," *Nature methods*, vol. 10, no. 6, pp. 563–569, 2013.

[13] K. Berlin, S. Koren, C.-S. Chin, J. P. Drake, J. M. Landolin, and A. M. Phillippy, "Assembling large genomes with single-molecule sequencing and locality-sensitive hashing," *Nature biotechnology*, vol. 33, no. 6, pp. 623–630, 2015.

[14] L. Salmela, R. Walve, E. Rivals, and E. Ukkonen, "Accurate self-correction of errors in long reads using de Bruijn graphs," *Bioinformatics*, vol. 33, no. 6, pp. 799–806, 2016.

[15] E. Haghshenas, F. Hach, S. C. Sahinalp, and C. Chauve, "CoLoRMap: Correcting Long Reads by Mapping short reads," *Bioinformatics*, vol. 32, no. 17, pp. i545–i551, 2016.

[16] S. Koren, M. C. Schatz, B. P. Walenz, J. Martin, J. T. Howard, G. Ganapathy, Z. Wang, D. A. Rasko, W. R. McCombie, E. D. Jarvis *et al.*, "Hybrid error correction and de novo assembly of single-molecule sequencing reads," *Nature biotechnology*, vol. 30, no. 7, pp. 693–700, 2012.

[17] M. Pop, A. Phillippy, A. L. Delcher, and S. L. Salzberg, "Comparative genome assembly," *Briefings in bioinformatics*, vol. 5, no. 3, pp. 237–248, 2004.

[18] K. F. Au, J. G. Underwood, L. Lee, and W. H. Wong, "Improving PacBio long read accuracy by short read alignment," *PloS one*, vol. 7, no. 10, p. e46679, 2012.

[19] G. Miclotte, M. Heydari, P. Demeester, S. Rombauts, Y. Van de Peer, P. Audenaert, and J. Fostier, "Jabba: hybrid error correction for long sequencing reads," *Algorithms for Molecular Biology*, vol. 11, no. 1, p. 10, 2016.

[20] L. Salmela and E. Rivals, "LoRDEC: accurate and efficient long read error correction," *Bioinformatics*, vol. 30, no. 24, pp. 3506–3514, 2014.

[21] Q. Zou, Q. Hu, M. Guo, and G. Wang, "HAlign: Fast multiple similar DNA/RNA sequence alignment based on the centre star strategy," *Bioinformatics*, vol. 31, no. 15, pp. 2475–2481, 2015.

[22] A. Bankevich, S. Nurk, D. Antipov, A. A. Gurevich, M. Dvorkin, A. S. Kulikov, V. M. Lesin, S. I. Nikolenko, S. Pham, A. D. Prjibelski *et al.*, "SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing," *Journal of computational biology*, vol. 19, no. 5, pp. 455–477, 2012.

[23] Y. Peng, H. C. Leung, S.-M. Yiu, and F. Y. Chin, "IDBA–a practical iterative de Bruijn graph de novo assembler," in *Annual International Conference on Research in Computational Molecular Biology*. Springer, 2010, pp. 426–440.

[24] K. Sameith, J. G. Roscito, and M. Hiller, "Iterative error correction of long sequencing reads maximizes accuracy and improves contig assembly," *Briefings in bioinformatics*, vol. 18, no. 1, pp. 1–8, 2016.

[25] D. R. Zerbino and E. Birney, "Velvet: algorithms for de novo short read assembly using de Bruijn graphs," *Genome research*, vol. 18, no. 5, pp. 821–829, 2008.

[26] K. Salikhov, G. Sacomoto, and G. Kucherov, "Using cascading Bloom filters to improve the memory usage for de Brujin graphs," *Algorithms for Molecular Biology*, vol. 9, no. 1, p. 2, 2014.

[27] T. H. Ogden and M. S. Rosenberg, "Multiple sequence alignment accuracy and phylogenetic inference," *Systematic biology*, vol. 55, no. 2, pp. 314–328, 2006.

[28] J. A. Cuff and G. J. Barton, "Application of multiple sequence alignment profiles to improve protein secondary structure prediction," *Proteins: Structure, Function, and Bioinformatics*, vol. 40, no. 3, pp. 502–511, 2000.

[29] A. Dereeper, S. Audic, J.-M. Claverie, and G. Blanc, "BLAST-EXPLORER helps you building datasets for phylogenetic analysis," *BMC evolutionary biology*, vol. 10, no. 1, p. 8, 2010.

[30] R. C. Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[31] M. J. Chaisson and G. Tesler, "Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory," *BMC bioinformatics*, vol. 13, no. 1, p. 238, 2012.

Yuansheng Liu received the bachelor and masters degrees in computer science from Xiangtan University, Xiangtan, China, in 2012 and 2015, respectively. He is currently pursuing the Ph.D. degree with the University of Technology Sydney, Australia.

In 2015, he was a Research Assistant with the Department of Electronic Engineering, City University of Hong Kong for two months and as a Research Assistant with the Dalian University of Technology for five months. His research interests include signal processing techniques for bioinformatics and multimedia security.

Chaowang Lan received the bachelor and masters degrees both from Guangxi university in 2008 and 2015, respectively. He is currently a Ph.D. student at the University of Technology Sydney, Australia. His research interests include data mining and bioinformatics.

Michael Blumenstein is currently the Head of the School of Software in the Faculty of Engineering and IT at the University of Technology Sydney, Australia. He comes from Griffith University in Queensland where he has accumulated over a decade of experience in leadership roles including portfolio Dean (Research) of the Sciences Group and Head of the School of ICT.

He is a nationally and internationally recognised expert in the areas of automated Pattern Recognition and Artificial Intelligence, and his current research interests include Document Analysis, Multi-Script Handwriting Recognition and Signature Verification. He has published 174 papers in refereed books, conferences and journals. His research also spans various projects applying Artificial Intelligence to the fields of Engineering, Environmental Science, Neurobiology and Coastal Management.

Jinyan Li is a Professor of Data Sciene and the Bioinformatics Program leader at the Advanced Analytics Institute, Faculty of Engineering and IT, University of Technology Sydney (UTS), Australia. He has a Bachelor degree of Science (Applied Mathematics) from National University of Defense Technology (China), a Masters degree of Engineering (Computer Engineering) from Hebei University of Technology (China), and a PhD degree (Computer Science) from the University of Melbourne (Australia). He joined UTS in March of 2011 after ten years of research and teaching work in Singapore (Institute for Infocomm Research, Nanyang Technological University, and National University of Singapore).

He is passionate about research on protein bindng free energy prediction, conformational B-cell epitope prediction, PPIs, disease-RNA-gene tripartite, NGS data management, and RNA-sequencing data analysis. He also loves research on data mining algorithms and machine learning methods. He has published 100 journal articles and 80 conference papers, of which many are highly cited.