BIOINFORMATICS ORIGINAL PAPER Vol. 27 no. 21 2011, pages 2957–2963
doi:10.1093/bioinformatics/btr507

Genome analysis Advance Access publication September 7, 2011

FLASH: fast length adjustment of short reads to improve genome assemblies

Tanja Magoč* and Steven L. Salzberg

McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University School of Medicine, Baltimore, MD 21205, USA

Associate Editor: Martin Bishop

ABSTRACT

Motivation: Next-generation sequencing technologies generate very large numbers of short reads. Even with very deep genome coverage, short read lengths cause problems in *de novo* assemblies. The use of paired-end libraries with a fragment size shorter than twice the read length provides an opportunity to generate much longer reads by overlapping and merging read pairs before assembling a genome.

Results: We present FLASH, a fast computational tool to extend the length of short reads by overlapping paired-end reads from fragment libraries that are sufficiently short. We tested the correctness of the tool on one million simulated read pairs, and we then applied it as a pre-processor for genome assemblies of Illumina reads from the bacterium *Staphylococcus aureus* and human chromosome 14. FLASH correctly extended and merged reads >99% of the time on simulated reads with an error rate of <1%. With adequately set parameters, FLASH correctly merged reads over 90% of the time even when the reads contained up to 5% errors. When FLASH was used to extend reads prior to assembly, the resulting assemblies had substantially greater N50 lengths for both contigs and scaffolds.

Availability and Implementation: The FLASH system is implemented in C and is freely available as open-source code at http://www.cbcb.umd.edu/software/flash.

Contact: t.magoc@gmail.com

Received on June 14, 2011; revised on August 25, 2011; accepted on August 31, 2011

1 INTRODUCTION

Thanks to the rapidly dropping cost of DNA sequencing technologies, *de novo* whole-genome sequencing (WGS) projects are generating very deep coverage of new genomes. However, even with the high coverage that is produced by these technologies, and despite dramatic improvements in genome assembly algorithms (Gnerre *et al*., 2011; Li *et al*., 2010), the short read lengths produced by next-generation technologies present a significant barrier to reconstructing a genome from WGS data. Any increase in read length will have a significant, positive impact on the quality of genome assemblies.

Several current protocols generate sequences from both ends of a library of DNA fragments. If the fragments are shorter than twice the read length, the resulting paired-end reads will overlap. For example, a project might use 175 bp libraries with 100 bp reads. This type of library presents an opportunity to extend the reads' length by overlapping and merging paired-end reads before using these reads in the assembly.

In this article, we present FLASH (Fast Length Adjustment of SHort reads), a software tool to find the correct overlap between paired-end reads and extend the reads by stitching them together. We tested the correctness of the tool on one million simulated 100 bp paired-end reads and found it to be highly reliable. We then used real data to measure the effect of this merging procedure on the quality of an assembly. We also compared the speed and accuracy of FLASH to that of SHERA (Rodrigue *et al*., 2010), a related tool that also extends read length by finding the overlap between paired-end reads.

2 METHODS

FLASH requires as input a fastq library of paired-end reads in which at least some of the reads overlap the read generated from the opposite end of the same DNA fragment. It processes each read pair separately and searches for the correct overlap between the paired-end reads. When the correct overlap is found, the two reads are merged, producing an extended read that matches the length of the original DNA fragment from which the paired-end reads were generated.

To find the correct overlap, FLASH considers every possible legal overlap between paired-end reads, where a legal overlap is defined as any ungapped alignment between two reads such that at least *min-olap* bases overlap one another. We chose to allow only ungapped alignments because with the Illumina sequencing platforms, which are the primary focus of our experiments, insertions and deletions are very rare. The overall flow of the algorithm is as follows:

(1) Align the pair of reads so that they overlap completely; e.g. by the full length of the shorter read.

(2) Repeat while the overlap is longer than *min-olap*:

   (a) Calculate the overlap length. If an 'N' occurs in the overlapping region, it is not counted towards the overlap length.

   (b) Calculate the score for the overlap as the ratio between the number of mismatches and the overlap length, ignoring N's.

   (c) If the score of the overlap is smaller than the score of the best overlap, save it as the new best overlap.

   (d) If the score is equal to the best previous score:
      (1) Calculate the average quality value of all mismatches in the overlap.
      (2) If the average quality value is smaller than the average quality value of mismatches in the best overlap, save the current overlap as the best overlap.

   (e) Slide the reads apart by one base, reducing the overlap by one.

*To whom correspondence should be addressed.

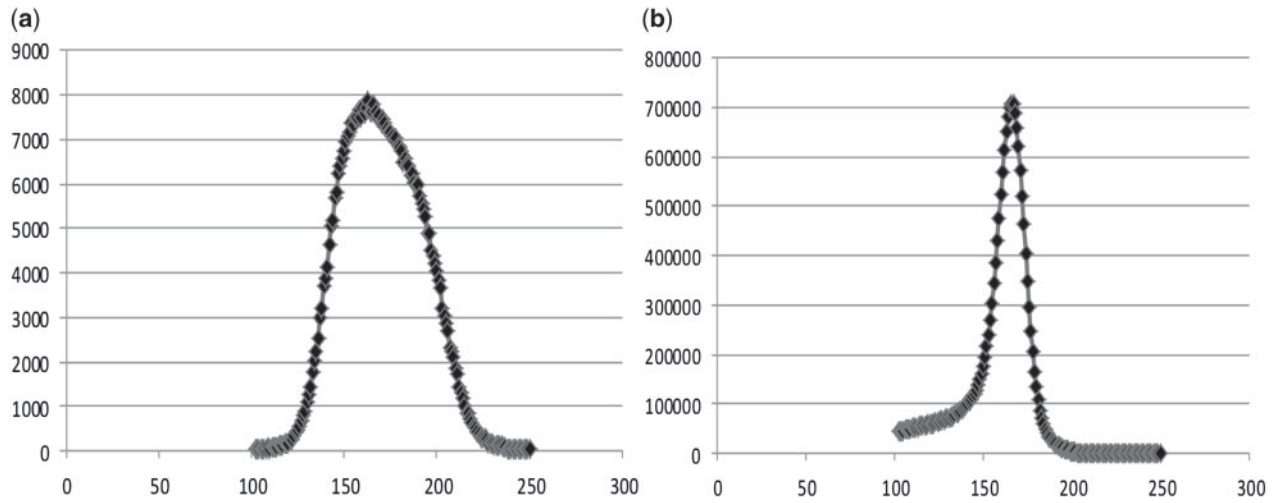

Fig. 1. Distribution of fragment lengths. The horizontal axis shows the fragment length, and the vertical axis shows the number of fragments of a given length. (**a**) Staphylococcus aureus. (**b**) Human chromosome 14.

(3) Compare the score of the best overlap to the mismatch threshold. If the score is bigger than the mismatch threshold, report that no good overlap was found; otherwise, return the best overlap.

FLASH scores every possible legal overlap between paired-end reads. Because a legal overlap is defined here as an ungapped alignment between two reads that overlap by at least *min-olap* bases, the maximum number of overlaps that are considered for each pair of reads is $O_{\max} = R - m$, where R is the read length and m is the minimum overlap. The number of overlaps considered might be smaller than $O_{\max}$ if the reads contain a large number of N's.

By default, the *min-olap* parameter is set to 10 bp. In our experiment, lower values of *min-olap* resulted in many incorrectly extended reads, because shorter overlaps often occur by chance in large WGS datasets, especially when mismatches are allowed. Higher values of *min-olap* will reduce bad merges further, but will make the system miss too many true overlaps. The trade-off between sensitivity and precision for different *min-olap* values is shown in Figure 6, described in Section 3.

To find the best overlap for a pair of reads, we calculate the score for each overlap as the ratio between the number of mismatches in the overlapping region and the length of that region. If an 'N' occurs at any position in any read, that position is ignored and not counted towards either mismatches or total overlap length. We select the overlap with the lowest ratio as the best overlap for a given read pair.

Assuming the DNA fragment size selection step was done with relatively tight controls, we expect the length of the overlap between paired-end reads within a library to be normally distributed, with a mean determined by the fragment size F and the read length R as $2R - F$. For our two sample datasets from *Staphylococcus aureus* and human chromosome 14, we were able to compute the true length empirically by mapping read pairs against the genomes. Figure 1 shows the distribution of fragment lengths. The bacterial fragments had a mean length of 170 bp while the human fragments were slightly shorter, ~165 bp. In the case of human data, the distribution is much tighter than in the case of bacterial data.

Considering that the typical Illumina error distribution has more errors towards the 3′-end of the read, we expect a higher ratio of true sequencing errors to occur in shorter overlaps, because both paired-end reads will only overlap in the regions with the highest error rate. This will lead any simple scoring scheme to prefer longer overlaps, which might not always be correct.

To determine whether a long overlap is better than a shorter one, we define *max-olap* to be the maximum length of the overlap expected in 90% of read pairs for a given read length and fragment size. For all overlaps longer than *max-olap*, we calculate their score as the ratio between the number of mismatches and *max-olap* rather than the actual overlap length. The parameter *max-olap* is set to 70 by default, which we found to work well for 100-bp Illumina reads from 180 bp fragments. Note that for the *S.aureus* data, 70 was too short because the fragment length was shorter than expected, and because the wide distribution of lengths resulted in many pairs overlapping by >70 bp. The algorithm's strong performance despite this deviation from the input fragment length indicates that the heuristic scoring function is robust.

The result of this scheme is that we sometimes prefer overlaps shorter than the maximum overlap detected, even if the mismatch to overlap ratio is greater for the shorter overlap. However, if an overlap longer than *max-olap* is substantially better than any shorter overlap, we select the longer overlap.

Because fragment lengths vary considerably, some pairs of reads will fail to overlap or will overlap by too few bases to be detected; e.g. for fragments >190 bp, our method will not detect overlap between two 100 bp reads. To distinguish between low-quality overlapping reads and non-overlapping reads, we created a mismatch ratio threshold, *mr*, which defines the maximum proportion of mismatches that we allow in any overlapping region. This value, set to 0.25 by default, can be increased to produce more aggressive read merging, or decreased to produce more conservative merging. The impact of different mismatch ratios on the accuracy of read merging is shown in Figure 5. As the figure shows, with a 1% error rate in reads, over 99% of read pairs are correctly processed when $0.2 \leq mr \leq 0.3$. We selected the median of this range (0.25) as the default value.

If the best overlap found between two paired-end reads has a mismatch ratio less than or equal to *mr*, the paired-end reads are merged. For each position in the overlapping region where the reads disagree, the algorithm chooses the base with the higher quality value. FLASH outputs a fastq file containing the extended reads, where each base in the overlapping region is assigned the quality value of the base with the higher quality where the reads agree, and a score of 2 where they disagree. FLASH also outputs two fastq files containing paired-end reads for which no good overlap was found. These unmerged reads can be used in the assembly as a normal paired-end fragment library.

3 RESULTS AND DISCUSSION

We tested the correctness of FLASH using both simulated and real data, and compared it for speed and correctness to SHERA

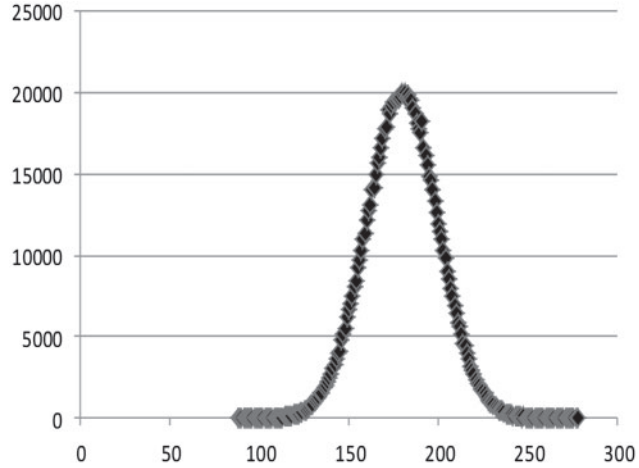

Fig. 2. Distribution of fragment lengths used to generate paired-end reads. The horizontal axis shows the fragment length, and the vertical axis shows the number of fragments of a given length.

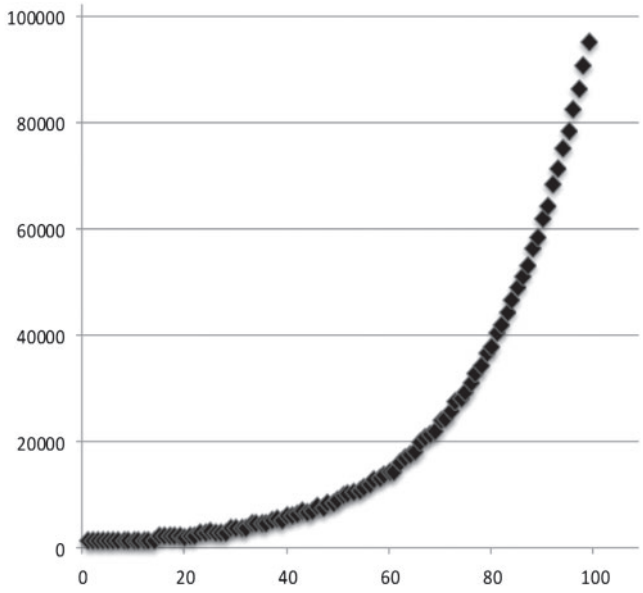

Fig. 3. Number of errors in the 1% error rate sample for each position in a read. The horizontal axis shows the read position, and the vertical axis shows the total number of errors at each position summed over the entire set of 1 000 000 pairs.

(Rodrigue *et al*., 2010), the only other existing tool that combines paired-end reads from short fragment library. We also measured how much FLASH affected the genome assemblies of *S.aureus* and human chromosome 14.

3.1 Simulated data

We simulated 1 000 000 pairs of 100 bp long reads. The paired-end reads were generated from fragments with a mean length of 180 bp, normally distributed with an SD of 20 bp. Error-free reads were generated using *wgsim* from the SAMtools package (Li *et al*., 2009). We inserted errors at a rate of 1, 2, 3 and 5% with an increasing probability of errors towards the end of reads. [The probability of error is defined by $y = a(1.05)^x$, where x is the read position and a and y are based on the error rate and read length.] In the simulated data, 16.4% of fragments had length ≥ 200, 5.5% had paired-end reads overlapping by < 5 bp, and an additional 8% had paired-end reads overlapping by < 10 bp. Thus, overall 30% of pairs overlapped by too little to be detected with default settings. A very small fraction of fragments ($< 1\%$) were <100 bp. The distribution of fragment lengths is shown in Figure 2, while the distribution of errors in the 1% error rate sample is shown in Figure 3. The simulated reads used in this data are available at the FLASH website.

When evaluating FLASH, we considered the trade-off between correct and incorrect merges of paired reads (Figs 5 and 6). We assume that two 100 bp reads should have been merged if they were generated from a fragment that was 100–190 bp in length, because we required a minimum overlap of 10 bp. Results for FLASH and SHERA for simulated reads with error rates ranging from 0% to 5% are shown in Table 1.

In the table, correct merges refers to the number of read pairs correctly overlapped and merged together (Fig. 4a). A merge is considered correct if a fragment of the correct length is produced, even if the wrong base is selected as the consensus base at one of the overlapping positions. (The only time there is a choice is when the two reads disagree at, a given site, in which case FLASH chooses the base with the higher quality value, breaking ties randomly.)

Correct non-merges are paired-end reads that were not merged together and that were generated from fragments that could not be extended (Fig. 4d); i.e. they were <100 bp or >190 bp. A total of 299 563 pairs in each simulated sample met this criterion.

Incorrect non-merges are pairs that were not merged even though they were derived from fragments between 100 bp and 190 bp in length (Fig. 4b). And finally, incorrect merges are read pairs that were merged together and where either (i) the reads were too far apart and should not have been merged (Fig. 4e) or (ii) the resulting extended read is the wrong length (Fig. 4c).

FLASH processed error-free reads correctly 99.7% of the time, with a very small number of false merges (2644) and just 10 cases where the system failed to merge a pair that it should have. In contrast, SHERA merged only 91.6% of the pairs correctly, with dramatically higher numbers of reads that were either merged incorrectly or failed to merge. For FLASH, the incorrect merging of reads results from two sources. First, if there are two distinct perfect overlaps between a pair of reads, FLASH selects the longer overlap, which on rare occasions results in an incorrect overlap. Second, because FLASH allows 25% of bases in the overlapping region to be mismatches, on rare occasions an apparent overlap comes from fragments >190 bp that do not actually overlap.

With a 1% error rate, which is approximately the error rate of current Illumina sequencers, FLASH correctly handles the same overall percentage of all pairs, with slightly fewer incorrect merges but more incorrect non-merges. SHERA also performed similarly to its results on error-free data, and again had far more incorrectly processed pairs than FLASH.

For reads with 2% error, FLASH still processed >98% of the pairs correctly, but the number of pairs that it failed to merge increased significantly. Increasing the maximum mismatch rate from 0.25 (the default) to 0.3 decreased these failed merges from 14 058 to 4410 (see the column labeled FLASH-0.3 under 2% error.) Thus, if we know that a dataset has a higher error rate, then we can adjust FLASH accordingly to improve its performance.

Table 1. Results of both systems on simulated reads

		Correct merge	Correct non-merge	Total correct (%)	Incorrect merge	Incorrect non-merge	Total incorrect (%)
Error-free	FLASH	700 347	296 999	99.73	2644	10	0.27
	SHERA	637 541	278 648	91.62	21 250	62 551	8.38
1% error	FLASH	699 177	297 669	99.68	2017	1137	0.32
	SHERA	629 166	279 829	90.90	20 106	70 899	9.10
2% error	FLASH	686 169	298 124	98.43	1649	14 058	1.57
	FLASH-0.3	695 797	294 809	99.06	4984	4410	0.94
	SHERA	617 247	280 922	89.82	19 044	82 787	10.18
3% error	FLASH	649 094	298 483	94.76	1393	51 030	5.24
	FLASH-0.35	690 561	292 390	98.30	7650	9399	1.70
	SHERA	602 232	281 873	88.41	18 138	97 757	11.59
5% error	FLASH	480 145	298 912	77.91	1214	219 729	22.09
	FLASH-0.35	641 346	295 108	93.65	5733	57 813	6.35
	SHERA	563 706	283 369	84.71	17 032	136 093	15.29

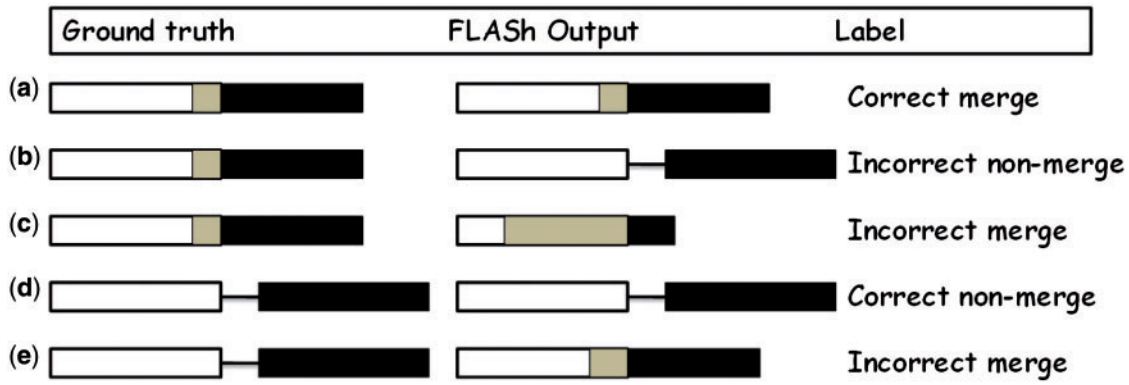

Fig. 4. Possible outcomes of FLASH for a pair of reads from opposite ends of the same fragment. The two reads are shown in white and black, and the grey region represents their overlap. For overlapping reads, FLASH can merge the pair correctly as shown at the top, or it can fail in two ways: either by failing to merge them or by creating the wrong length overlap. If the reads do not overlap, the correct output will leave them unchanged (a 'non-merge').

At even higher error rates of 3 and 5%, FLASH still handled a large majority of pairs correctly, but the numbers dropped dramatically at 5%. Increasing the mismatch ratio to 0.35 for both these trials increased the correct performance of FLASH, and for the 5% error data the results were quite good, improving from 78% to almost 94% of read pairs. Overall, FLASH was superior to SHERA on nearly all these simulated datasets.

3.2 Impact of different parameters on the accuracy of FLASH

Next we evaluate the impact of two key parameters on the accuracy of the read merging procedure. The tests below were performed on simulated reads with a 1% error rate, keeping all parameters constant except for the one being evaluated.

3.2.1 Mismatch ratio Figure 5 shows how the mismatch ratio parameter affects the number of correctly and incorrectly merged reads. As expected, increasing the allowed mismatch ratio increases the number of correctly merged read pairs, but at the expense of also increasing the number of incorrectly merged pairs. It is also clear from the graph that as the ratio drops <0.2, the number of correctly merged pairs drops very steeply, making the optimal value for these data, with 1% error, to be a mismatch ratio between 0.2 and 0.3.

3.2.2 Minimum overlap length Figure 6 shows the number of correctly and incorrectly merged read pairs for different values of the minimum overlap length parameter. As we decrease the minimum required overlap length, the number of correctly extended reads steadily increases, as expected. Interestingly, as the overlap drops below 15 and then to 10 bp the number of correctly merged pairs levels off, while the errors increase rapidly.

3.3 Time requirements

FLASH currently runs only in single threaded mode, but it is very fast especially for small to moderate datasets. We compared FLASH and SHERA on one million pairs of 100 bp long reads. We measured

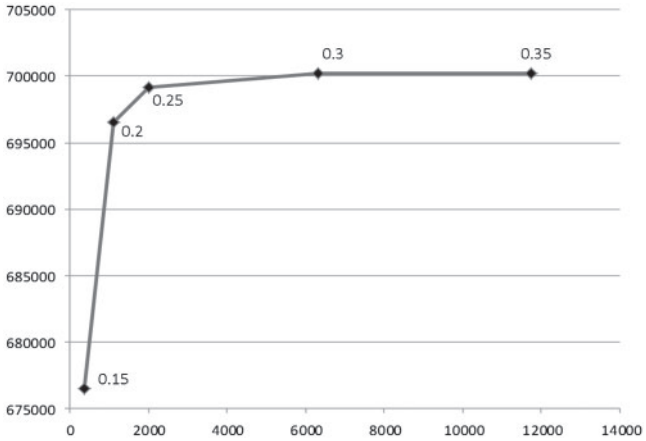

Fig. 5. Impact of the mismatch ratio parameter on correctness of the read merging algorithm. The horizontal axis shows the number of incorrectly merged read pairs, and the vertical axis shows the number of correctly merged read pairs. The mismatch ratio parameter is shown at each point along the graph.

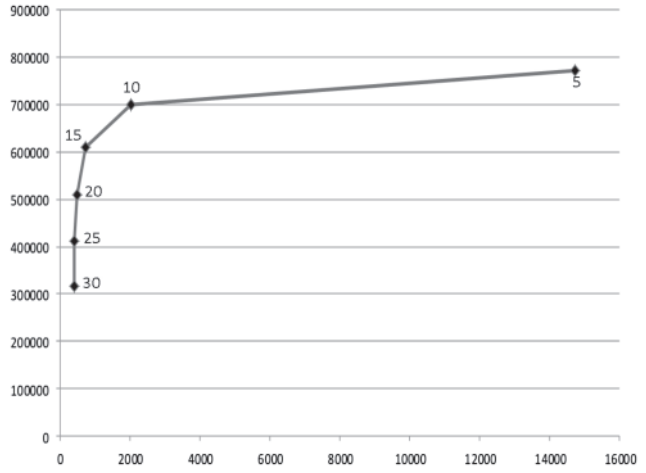

Fig. 6. Impact of the minimum overlap parameter on correctness of the read merging algorithm. The horizontal axis shows the number of incorrectly merged read pairs, and the vertical axis shows the number of correctly merged read pairs. The minimum overlap value (in base pair) is shown on the graph.

performance on two computers: a six-core 2.4 GHz AMD Opteron server with 256 GB of RAM, and a dual-core Intel Xeon 3.0 GHz desktop computer with 2 GB of RAM. Run times are summarized in Table 2.

As Table 2 shows, FLASH is far faster than SHERA. The most likely source of the speedup is that FLASH is implemented in C, while SHERA uses Perl, an interpreted language. The overall run time for FLASH is linearly proportional to read length times the number of reads.

3.4 Read merging and genome assembly

FLASH essentially creates longer reads from all the read pairs that it correctly merges. Longer read lengths should have a significant positive impact on whole-genome assembly, and therefore we wanted to test FLASH on real data by using it as a pre-processor

Table 2. Time requirements (in seconds) to process one million pairs of 100 bp long reads on two computer systems

Computer	Dual core, 3.0 GHz	six core, 2.4 GHz
FLASH	129	120
SHERA	14 200	8500

Table 3. Genome assemblies of *S.aureus*

	Original assembly	FLASH	SHERA
Total contig size (Mb)	2.91	2.94	2.96
Contig N50 size (kb)	1.45	8.40	6.04
Contig maximum (kb)	8.18	36.07	23.85
Scaffold N50 (kb)	2.07	8.80	6.40
Scaffold maximum (kb)	11.23	36.07	23.85

The N50 size is the size at which 50% of the genome is contained in contigs (or scaffolds) of size N50 or larger. N50 values were calculated based on the known genome size of 2 872 915 bp. The total contig size is the number of bases contained in contigs that are at least 100 bp long.

for two assemblers: CABOG (Miller *et al*., 2008), which is a recent version of the Celera Assembler, and SOAPdenovo (Li *et al*., 2010). For the two genomes used in our tests, a bacterium and the human genome, the true assembly is known, which allowed us to evaluate the correctness of merged read pairs as well as the correctness of the assemblies.

3.4.1 Assembly of Staphylococcus aureus For our first test, we used short-read data from *S.aureus*, which was sequenced to deep coverage with Illumina sequencing technology by the Broad Institute (MacCallum *et al*., 2009). The data comprise two paired-end libraries, each of which we randomly sampled to 45X coverage. The first fragment library (SRA accession SRX007714) contains 647 052 pairs with a fragment size of 180 bp and read length of 101 bp. The second library (SRX007711) contains 1 747 035 pairs with a fragment size of 3.5 kb and read length of 37 bp. The reads are available from the NCBI Sequence Read Archive or from `http://gage.cbcb.umd.edu/data`.

We performed error correction on these reads using Quake (Kelley *et al*., 2010) with a *k*-mer size of 18. Quake discards reads that have too many errors, which results in some reads losing their 'mate'. We used these reads as an unpaired library, along with the two paired-end error corrected libraries to assemble *S.aureus* using SOAPdenovo. Note that we could not run CABOG on this dataset, because that assembler requires a minimum read length of 64 bp. The results from the initial SOAPdenovo assembly are shown in Table 3 in the *Original* column.

Next, we used FLASH to merge read pairs from the 180 bp fragment library, which resulted in 369 496 merges. With Illumina paired-end sequencing, the second read in each pair is typically of much lower quality, which probably explains why FLASH could not merge a higher percentage of the reads. Figure 1 shows that the fragment lengths were ∼170 bp. We retained the read pairs that were not merged and used them in the assembly as a paired-end library with insert size of 180 bp. As in the first assembly, we ran

Table 4. Results of merging read pairs on *S.aureus* data

	FLASH	SHERA
Correct merge	347 833	346 141
Correct non-merge	239 510	174 350
Total correct (%)	90.8	80.4
Incorrect merge	21 663	55 521
Incorrect non-merge	38 046	71 040
Total incorrect (%)	9.2	19.6

Table 5. Genome assembly of human chromosome 14 using SOAPdenovo

	Original assembly	FLASH assembly
Total contig size (Mb)	90.58	94.29
Contig N50 (kb)	3.48	3.78
Maximum contig length (kb)	37.02	37.66
Scaffold N50 (kb)	313.48	399.84
Maximum scaffold length (Mb)	1.49	2.53

N50 sizes are calculated based on the size of Hs14 in build 19 of the human reference genome, which is 88 289 540 bp.

Quake to correct reads. We then ran SHERA to merge reads from the 180 bp fragment library, and as before, we used Quake to correct reads before assembly.

As Table 3 shows, using FLASH to merge reads produced much larger contigs and scaffolds than both the original assembly and the assembly that used SHERA to merge reads. The improvements over the original assembly were particularly dramatic: the contig N50 size increased >5-fold, and scaffold N50 size increased 4-fold.

To check the correctness of the read merging algorithm, we used Bowtie (Langmead *et al*., 2009) to map the merged and non-merged read pairs to the finished *S.aureus* genome. As shown in Table 4, FLASH correctly processed 90.8% of the reads versus 80.4% for SHERA. The larger number of incorrectly merged pairs is probably the explanation for the poorer genome assembly results (Table 3).

We also compared the accuracy of the assemblies generated by SOAPdenovo with and without merging read pairs. To evaluate accuracy, we used the nucmer program from the MUMmer package (Kurtz it et al., 2004) to map all contigs at least 200 bp long to the reference genome. We considered a contig correct if it mapped with at least 98% identity over 98% of its length. For both assemblies, six contigs failed to map correctly. Thus, there was no difference in correctness while the FLASH-assisted assembly contained far larger contigs.

3.4.2 Assembly of human chromosome 14 For the second assembly comparison, we used three paired-end libraries from the human sequence NA12878 (SRA024407) sequenced at the Broad Institute (Gnerre *et al*., 2011). The reads were selected by using Bowtie to map the reads from NA12878 onto the entire human genome, and then selecting only those pairs that mapped to chromosome 14. The mapping yielded 18 252 400 pairs in the 180 bp fragment library, 11 334 704 pairs in the 3000 bp library and 1 202 532 pairs in the 35 kb library. All reads are 101 bp long. The chromosome 14 reads are available from http://gage.cbcb.umd.edu/data/index.html.

Table 6. Genome assembly of human chromosome 14 using CABOG

	Original assembly	FLASH assembly
Total contig size	86.28 Mb	86.76 Mb
Contig N50	41.54 kb	90.67 kb
Maximum contig length	276.78 kb	539.38 kb
Scaffold N50	360.87 kb	1.10 Mb
Maximum scaffold length	2.13 Mb	4.05 Mb

Table 7. Accuracy of contigs in assemblies of human chromosome 14

	SOAPdenovo only	FLASH plus SOAPdenovo	CABOG only	FLASH plus CABOG
Total contigs	46 264	42 751	3674	1899
Correct contigs	45 215	41 779	3184	1588
Incorrect contigs	1049	972	490	311

All contigs <200 bp were used for the evaluation.

As with the assembly of *S.aureus*, we used Quake with a *k*-mer size of 18 to correct all reads, and used the corrected reads from three paired-end libraries in the assembly by SOAPdenovo and CABOG.

We then merged reads in the 180 bp library using FLASH, which yielded 17 047 292 extended reads. Note that this was a far higher percentage of the library than were merged in the *S.aureus* data; this is likely due to the fact that the distribution of fragment lengths in the human data was much tighter (Fig. 1), making more fragment lengths fall within the detectable overlap range of FLASH.

The assembly results are shown in Tables 5 and 6. As with *S.aureus*, the assembly that used merged reads from FLASH outperformed the assembly that used the original reads in every category regardless of the assembler used to assemble the genome. The CABOG assembly had far larger contigs and scaffolds than the SOAPdenovo assembly, and the improvements from FLASH were also much more striking: the contig N50 size more than doubled, and the scaffold N50 size tripled.

We then mapped the merged and non-merged reads to the reference genome to measure how many had been correctly processed. FLASH's results were very similar to those for *S.aureus*, handling 91% of pairs correctly as before. Failure to merge reads again accounted for a large majority of its mistakes: 1.1 million pairs were left separate although they overlapped sufficiently for merging. Over 16 million pairs were correctly merged while 572 622 pairs were merged incorrectly.

Finally, we compared the accuracy of the assemblies that used FLASH as a preprocessor to those that did not. Note that these are *de novo* human assemblies, constructed without use of the reference human genome. Comparative assemblies, which typically have far larger contigs than *de novo* assemblies, might not show as great a difference if FLASH was used.

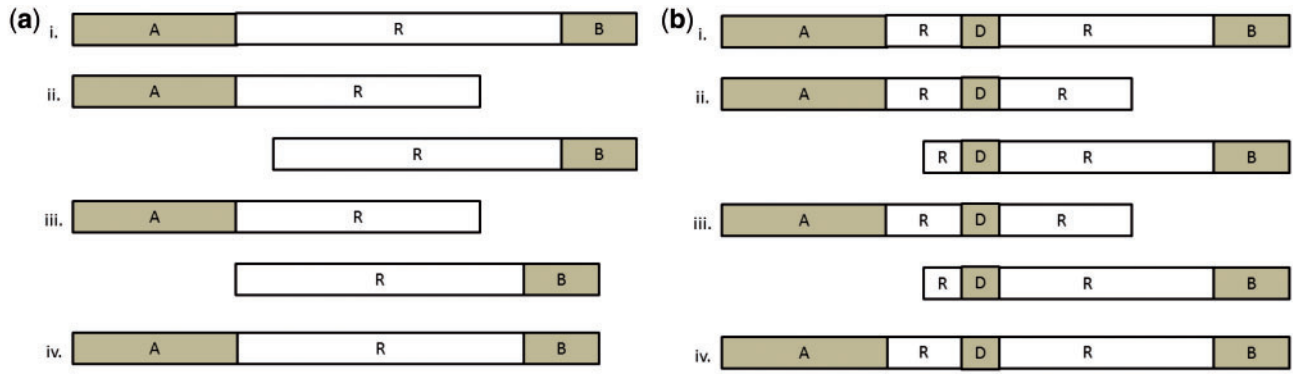

Fig. 7. Illustration of how exact tandem repeats might be collapsed. A and B represent unique sequences flanking R, which is a tandem repeat. On the left (**a**), R contains multiple identical copies of a the same subsequence. At the top (i) is the original fragment, and just below that (ii) are the two overlapping reads sequenced from each end. The best overlap on the left (iii), shows that the reads overlap too much, which collapses R, eliminating one or more copies of the repeat (iv). On the right (**b**), the copies are not identical. D is a sequence (as short as one base) that makes one tandem copy different from the others. As a result, the best overlap (iii) produces the correctly merged reads (iv).

As above, we considered a contig to be correct if 98% of its length aligned to the genome with at least 98% identity. Table 7 shows the results for assemblies by both SOAPdenovo and CABOG.

As shown in Table 7, using FLASH as a preprocessor for genome assembly not only improved the assembly by producing fewer, larger contigs, but it also reduced the number of erroneous contigs for both SOAPdenovo and CABOG. This also shows that FLASH is not limited to just one assembler, and suggests that it would be a useful adjunct to standard assembly pipelines.

3.5 Difficulties with tandem repeats

Merging reads from opposite ends of a short fragment can run into problems when the fragment contains short tandem repeats. If the overlapping regions contains nothing else, the two reads might be merged too aggressively, collapsing the repeat region into fewer copies than are actually present in the fragment. While exact tandem repeats might cause problem for FLASH, non-exact tandem repeats should be resolved easily.

The Figure 7 illustrates the problem. Figure 7a (i) shows the original fragment, with unique sequences A and B flanking a region R that contains multiple copies of a short tandem repeat. The left part of the figure illustrates the situation when the tandem repeats are identical, while the right-hand side shows what happens when the repeats contain some variation. As shown in Figure 7a (iii), the best overlap might be too short when the repeats are exact, leading FLASH to collapse the repeat region slightly. This problem only arises when the repeats are exact. If R is shorter than the read length, then some reads will contain R entirely, allowing a genome assembler to resolve it correctly.

4 CONCLUSION

The short read lengths generated by next-generation sequencing technologies are the biggest challenge in assembling genomes into large, near-complete sequences. To tackle this problem, we developed FLASH, a highly accurate, very fast tool to merge pairs generated by sequencing both ends of short fragment libraries. FLASH can be used for any paired-end sequence data in which the paired reads overlap, including RNA-seq data. As shown in our experiments, merging paired reads with FLASH can have a significant positive impact on the quality of genome assemblies.

Funding: National Institutes of Health under grants (R01-LM006845 and R01-GM083873) in part.

Conflict of Interest: none declared.

REFERENCES

Gnerre,S. *et al.* (2011) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl Acad. Sci. USA*, **108**, 1513–1518.

Kelley,D.R. *et al.* (2010) Quake: quality-aware detection and correction of sequencing errors. *Genome Biol.*, **11**, R116.

Kurtz,S. *et al.* (2004) Versatile and open software for comparing large genomes. *Genome Biol.*, **5**, R12.

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

Li,H.; 1000 Genome Project Data Processing Subgroup (2009) The sequence alignment/map (SAM) format and SAMtools. *Bioinformatics*, **25**, 2078–2079.

Li,R. *et al.* (2010) De novo assembly of human genomes with massively parallel short read sequencing. *Genome Res.*, **20**, 265–272.

MacCallum,I. *et al.* (2009) ALLPATHS2: small genomes assembled accurately and with high continuity from short paired reads. *Genome Biol.*, **10**, R103.

Miller,J.R. *et al.* (2008) Aggressive assembly of pyrosequencing reads with mates. it *Bioinformatics*, **24** , 2818–2824.

Rodrigue,S. *et al.* (2010) Unlocking short read sequencing for metagenomics. *PloS One*, **5**, e11840.