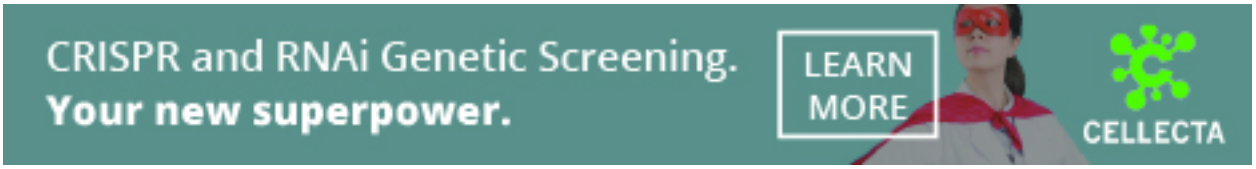
Short read fragment assembly of bacterial genomes

Mark J. Chaisson and Pavel A. Pevzner

Genome Res. 2008 18: 324-330 originally published online December 14, 2007
Access the most recent version at doi:10.1101/gr.7088808



Short read fragment assembly of bacterial genomes

Mark J. Chaisson[1] and Pavel A. Pevzner[2,3]

[1]*Bioinformatics Program, University of California San Diego, La Jolla, California 92093, USA;* [2]*Department of Computer Science and Engineering, University of California San Diego, La Jolla, California 92093, USA*

In the last year, high-throughput sequencing technologies have progressed from proof-of-concept to production quality. While these methods produce high-quality reads, they have yet to produce reads comparable in length to Sanger-based sequencing. Current fragment assembly algorithms have been implemented and optimized for mate-paired Sanger-based reads, and thus do not perform well on short reads produced by short read technologies. We present a new Eulerian assembler that generates nearly optimal short read assemblies of bacterial genomes and describe an approach to assemble reads in the case of the popular hybrid protocol when short and long Sanger-based reads are combined.

In 2005, the first successful whole-genome shotgun sequencing project performed without using Sanger-based sequencing was completed (Margulies et al. 2006). The analytical method, pyrosequencing, was proposed in 1998 (Ronaghi et al. 1998) and has been incorporated into an industrial sequencing platform by 454 Life Sciences (www.454.com). While this was one of the original technologies for high-throughput sequencing, many other next-generation competing platforms, including the Solexa/Illumina 1G Genome Analysis System (www.illumina.com) and Applied Biosystems SOLiD Sequencing (www.appliedbiosystems.com), have recently emerged and have made their data available to many research groups. Other recent start-ups, including Helicos (www.helicosbio.com) and Complete Genomics (www.completegenomics.com), plan to release their sequencing data soon (see Metzker 2005 for an overview of emerging short read technologies). Although each technology is able to produce vast quantities of sequence information, in every case the underlying chemistry limits reads to very short lengths in comparison with Sanger-based reads: from over one hundred bases (454 Life Sciences) down to as little as 24 (Solexa/Illumina and Applied Biosystems). These technologies usher a new era of high-throughput short read (HTSR) sequencing.

The applications of HTSR sequencing have been largely limited, albeit with great success, to resequencing (Andries et al. 2005), gene expression (Kim et al. 2007), and genomic profiling (Barski et al. 2007), which all require a reference genome. Although many model organisms have been sequenced, the success of recent comparative genomics studies (The ENCODE Project Consortium 2007) will increase the demand for de novo assembly, which will be assisted by the assembly of HTSR sequences. Furthermore, the discovery of rampant structural polymorphisms in the human genome (Feuk et al. 2006) indicates that it is necessary to perform de novo assembly of even the same species to capture all variations.

Even before the release of the first draft bacterial genome assembly by 454 Life Sciences, we published a paper examining de novo assembly of short reads (Chaisson et al. 2004; see also Whiteford et al. 2005). In the absence of available data, we simulated reads from bacterial genomes with lengths and error rates of what the contemporary proof-of-concept short-assembly papers had cited. We came to the conclusions that while the high-coverage facilitates the error correction in reads prior to assembly, it will be difficult to assemble very large contigs without mate-pair information. Recently, 454 Life Sciences developed the Newbler assembler that is now a part of the software package distributed with 454 sequencing machines. Newbler is an excellent assembler that takes into account the specifics of pyrosequencing errors to generate accurate contigs.

The article by Margulies et al. (2006) has shown that short read assembly is indeed difficult; the assembly of *Streptococcus pneumoniae* had 255 long contigs (i.e., contigs longer than 500 bp), of which 11 were misassembled. Later, the proprietary 454 Newbler assembler was improved, and it now produces 253 long contigs (of total length 2000 kb) and no missassembled contigs. The *S. pneumoniae* genome contains 167 exact (unresolvable) repeats at least 120 bases long[4] that fragment the assembly into 504 contigs, of which 136 are longer than 500 bp. The difference between 253 and 136 illustrates the gap between the performance of the existing short read assemblers and the lower bound on the number of contigs an optimal assembler would produce. Our new assembler, EULER-SR, produces only 127 long contigs of total length 2001 kb, thus illustrating that it comes close to the "optimal" assembler (136 long contigs of total length 2091 kb).

The assembly of short reads is delayed by the inability to adapt older assembly techniques from previous sequencing projects. Many whole-genome assemblers are optimized for assembling the mate-paired long reads that until recently have been the sole type of data produced by large-scale sequencing centers. Programs such as Phrap (www.phrap.org) and PCAP (Huang et al. 2003) simply do not run on next-generation data, and ARACHNE (Jaffe et al. 2003) produces an overfragmented output in the absence of mate-pairs. Similarly, EULER+ assembler (Pevzner et al. 2004) does not scale well for high-coverage short read sequencing projects.

The Eulerian approach for assembly was proposed by Pevzner et al. (1989) in the context of sequencing by hybridization (SBH), which may be viewed as the very first short read sequencing technology.[5] In the pioneering paper, Idury and Waterman

[3]Corresponding author.
E-mail ppevzner@cs.ucsd.edu; fax (858) 534-7029.
Article published online before print. Article and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.7088808.

[4]Most reads in this study are shorter than 120 bases.

[5]The paper by Pevzner et al. (1989) illustrates some potential advantages of the Eulerian path approach over the "overlap-layout-consensus" approach for fragment assembly. For example, while the study by Pevzner et al. (1989) described a simple algorithm for constructing the SBH repeat graph, it is not immediately clear how to generalize the approaches in the studies by Huang et al. (2003) and Jaffe et al. (2003) for efficient construction of the repeat graph even in the simple case of SBH "reads."

18:324–330 ©2008 by Cold Spring Harbor Laboratory Press; ISSN 1088-9051/08; www.genome.org

(1995) extended the Eulerian approach to Sanger reads. Briefly, in Eulerian assembly, a de Bruijn graph is created with a vertex v for every l-tuple present in a set of reads, and an edge (v, v') is created if an $(l + 1)$-tuple exists in the reads that contains v as an l-prefix and v' as an l-suffix. An assembly corresponds to an Eulerian path (Cormen et al. 1995) through this graph. However, the practical applications of both Pevzner et al. (1989) and Idury and Waterman (1995) were hindered by high error rates of SBH and Sanger sequencing (circa 1995) that made the direct applications of the Eulerian approach difficult. In the Eulerian assembly described in Pevzner et al. (2001), sequencing errors create extra vertices and edges in this graph that require error correction procedures to improve the assembly quality.

More recently, Pevzner et al. (2004) proposed a new Eulerian assembler based on the notion of A-Bruijn graphs. The EULER+ assembler (Pevzner et al. 2004) deals with errors in reads by inducing vertices with ungapped alignments that allow mismatches, rather than the exact l-tuples in de Bruijn assembly, and by defining a set of graph-simplification techniques that remove erroneous edges. Large benchmarking experiments in Pevzner et al. (2004) demonstrated that the A-Bruijn approach compares well with Phrap and ARACHNE in assembling BACs and bacterial genomes. A similar approach was recently described by Myers (2005) and Medvedev et al. (2007). The A-Bruijn graph was recently used in various applications including multiple alignment (Raphael et al. 2004), finding composite repeats (Zhi et al. 2006), shotgun protein sequencing (Bandeira et al. 2007), and dissecting the history of segmental duplications in human genome (Jiang et al. 2007). However, the application of the A-Bruijn-based assembly for short read sequencing (with much higher coverage than in typical long-read sequencing projects) is difficult since the graph structure that is used scales with coverage and thus demands a large amount of memory. Therefore, the approach in Chaisson et al. (2004) has to be modified for high-coverage (and thus memory-intensive) short read assembly.

In this paper we describe how to modify the A-Bruijn technique from Pevzner et al. (2004) for short read assembly and present a memory-efficient de Bruijn-based approach that supports A-Bruijn-like graph correction operations. Our EULER-SR assembler substitutes the maximum spanning tree optimization of the A-Bruijn graph by the maximum branching optimization on de Bruijn graphs. We benchmark our new EULER-SR assembler on data generated by two 454 bacterial sequencing projects (with an average read length of ≈100 bp). Our assembly of 454 reads compares well with the proprietary 454 assembly tool Newbler and is the only assembler tool currently capable of assembling shorter reads such as those produced by the Solexa (we were able to produce nearly optimal BAC assembly with Solexa reads). In a difference from the Newbler assembler that outputs the set of unrelated contigs, EULER-SR outputs both the contigs and the repeat graph (Pevzner et al. 2001) of the assembled genome, thus connecting the contigs by repeats and directing finishing effort. In addition, this facilitates the use of mate-pairs when they become available, as recently announced by 454 Life Sciences and Illumina.

Currently, most sequencing centers utilize 454 reads by combining them with low-coverage Sanger-based reads for bacterial assembly and finishing (Goldberg et al. 2006). They first assemble only 454 reads using Newbler and later transform long Newbler contigs (longer than 500 bp) into overlapping pseudo-reads modeling typical Sanger reads. These pseudo-reads are further combined with real Sanger reads and subjected to a conventional assembly with programs like ARACHNE. This computational protocol is clearly not optimal: For example, even if Newbler was substituted by an "ideal" assembler, the valuable information about short contigs (shorter than 500 bp) would be lost. In particular, all valuable information about repeated regions shorter than 500 bp (covered by 454 reads) is not utilized in this hybrid approach. However, since Newbler is not designed to handle mate-pairs and since ARACHNE is not designed to handle short reads, no better solution is currently available. Below we show that EULER-SR is well suited for hybrid 454+Sanger approach to assembly and finishing.

Results

We analyzed 454 reads from *Escherichia coli* and *S. pneumoniae* (454 Life Sciences), and Solexa reads from a human BAC[6] (Solexa). Many HTSR sequencers produce raw data that are different from the traditional read traces found in Sanger sequencing, such as the "flowgram" reads produced by 454 or "color-space" reads produced by Applied Biosystems. It is possible to take advantage of the additional information in the raw read information, as done by Newbler assembler (454 Life Sciences); however, EULER-SR solely uses reads and quality files (if available) so that it is applicable to any sequencing technology. Below we demonstrate that EULER-SR compares well with 454 Newbler assembler and generates nearly optimal assembly of Solexa reads, despite the fact it does not take advantage of the peculiarities of the platform-dependent error models.

Error correction

Our approach begins by preprocessing reads to remove errors. When quality values are available, we trim or discard low-quality reads; however, typically our error correction routine will detect low-quality reads without quality values. Our method for error correction, as described in Methods, introduces the fewest changes possible in a read, such that every l-tuple in the read belongs to a set of l-tuples called a "spectrum," $\mathcal{S}$, or considers the read unfixable if too many changes are required. We iteratively apply our error correction as described in Methods to a set of reads, first setting $\mathcal{S}$ so that reads in regions of high coverage are corrected, and gradually increase the set of reads in which we fix errors by expanding the set $\mathcal{S}$.

We tuned the parameters for iteratively fixing errors on 454 *E. coli* reads. This data set has ~1.1 M reads for 27-fold coverage of the genome. The reads are relatively high quality on average, and only 0.5% of bases were trimmed using quality scores. We measure the rate of error correction by measuring the number of error-free reads, that is, reads that appear exactly in the genome (68% of the reads error-free). Our error correction routine divides the reads into two sets: a large set of corrected reads and a small set of nonfixable reads. The nonfixable set has ≈0.1 M reads (9% of all reads), and of the remaining corrected set, 99.6% reads are error-free (Table 1). Applying our error correction to the *S. pneumoniae* (without changing parameters) resulted in slightly lower (99.1%) but still high rate of error correction.

The bulk of EULER-SR time is taken by error correction; it takes ≈12 h to perform error correction on a single, 1.8-GHz workstation and only ≈1/2 h to assemble the genome. Fortu-

[6] *E. coli* is 4,639,675 bp long, *S. pneumoniae* is 2,160,837 bp long, and BAC is 173,427 bp long (chromosome 6, bases 30537344–30710771).

Table 1. The results of error correction on 454 GS20 reads

Genome	Total (error-free)	Reads, in thousands: Quality trimmed (error-free)	Reads, in thousands: Error corrected (error-free)
E. coli	1113 (757)	1113 (765)	1012 (1008)
S. pneumoniae	1063 (605)	1063 (652)	963 (954)

Error correction using SA was applied to each set of reads after trimming ends of reads with *phred* scores averaging <20 over five bases. Reads were mapped to the original genome by querying a compressed suffix array as described in Lippert et al. (2005).

nately, error correction represents an embarrassingly parallel application and only takes a matter of minutes on a modest cluster. EULER-SR provides support for parallelizing error correction on the Sun Grid Engine (SGE) cluster platform.

Fragment assembly of short reads

After correcting errors, we ran EULER-SR on the set of corrected reads. Our assembly procedure is a set of steps that first builds the de Bruijn graph, corrects errors in the graph, and transforms paths in the graph into assembled sequences. When constructing the de Bruijn graph, the l-tuple size should be chosen to be large enough to avoid excessive tangling of the graph that leads to shorter contigs, yet small enough to not fragment the genome since every $(l + 1)$-tuple must be covered by a read in the de Bruijn graph. We found that $l = 35$ removes most tangles at a cost of slightly fragmenting the assembly. To resolve the fragmentation, we "artificially" connect source and sink edges that share a smaller unique overlap at the last stage of the algorithm. We combined the error-corrected reads with the nonfixable reads that were discarded during error correction to assemble low-coverage regions.

The parameters for graph correction control the maximum girth (for bulge and whirl removal), maximum source or sink edge length to delete during erosion, and minimum edge multiplicity to detect chimeric edges. We tuned our parameters for assembling *E. coli* and used this to assemble *S. pneumoniae*. After error correction, all computations were performed on a desktop computer. We compared our assembly with ones produced by the 454 Newbler assembler (Table 2). For each assembly, we report the number of long contigs (i.e., contigs longer than 500 bp), the total length of long contigs, the N50 contig size (the size of the contig such that 50% of the assembly is contained in contigs of size N50 or greater), and the coverage using long contigs. Each of these statistics in isolation may be misleading, for example, an assembler that misjoins contigs may have the best parameters. However, neither the improved version of Newbler nor EULER-SR produced misjoined contigs, making these parameters comparable for different assemblers.[7]

Table 2. Summary of bacterial assemblies using 454 reads

Genome	Assembler	No. long contigs	Total length of long contigs (in kb)	N50 (in bases)
S. pneumoniae	EULER-SR	127	2001	32,619
	Newbler	253	2000	11,905
	Repeat graph	136	2091	36,004
E. coli	EULER-SR	199	4277	46,887
	Newbler	141	4531	60,757
	Repeat graph	94	4560	125,693

Reported are assemblies using EULER-SR and Newbler, and the repeat graph as a reference for the optimal assembly.
Number of long contigs: The number of contigs greater than 500 bases.
Total length of long contigs: The sum of the length of all long contigs.
N50: The size of the smallest contig such that 50% of the length of the genome is contained in contigs of size N50 or greater.

The theoretical limits of short read assembly may be evaluated by constructing the repeat graph of a genome with a minimum repeat length equal to the typical length of a read. The repeat graph corresponds to the condensed de Bruijn graph if an error-free read is generated at each position in the genome. Therefore, the repeat graph models an "optimal" assembler that can be benchmarked against the existing assemblers (Table 2). The optimal assembly of *E. coli* on reads of length 120 has 800 contigs with 94 long (i.e., longer than 500 bases) contigs covering 4560 kb. The optimal assembly of *S. pneumoniae* on reads of length 120 has 504 contigs with 136 long contigs covering 2091 kb. For *S. pneumoniae*, EULER-SR comes close to the optimal assembler with 127 long contigs covering 2001 kb. Newbler generated twice the number of larger contigs (253) with similar total size (2000 kb). We emphasize that Newbler uses the platform-dependent error model of 454 technology and has access to flowgrams reflecting the specifics of pyrosequencing errors. This may explain the deterioration in EULER-SR performance in the case of *E. coli* (199 long contigs covering 4277 kb) as compared with Newbler (141 long contigs covering 4531 kb). When Newbler is run on a FASTA data set that is the equivalent of only nucleotide sequences (an abuse of the intended usage), its performance degrades to 311 long contigs covering 4523 kb. This experiment demonstrates that Newbler is an excellent tool for assembling 454 reads (as it uses flow values to mitigate sequencing errors) that may not be easy to modify for other sequencing technologies. As 454 read quality is constantly improving, the additional data provided by 454 flowgrams is becoming less crucial. The difference in the error rates between different sequencing projects undertaken at different times/laboratories may explain a somewhat better EULER-SR performance on *S. pneumoniae* as compared with *E.coli*.

We have also assembled a human BAC using ≈1 million 27-nucleotide Solexa reads generated by the Joe Ecker laboratory at the Salk Institute (150× coverage). The Solexa technology is currently being optimized, and the error model of the Solexa platform remains poorly understood. In particular, the error rates of Solexa reads are position-dependent and highly variable across different machines and even across different runs of the same machine (Alla Lapidus, pers. comm.). As a result, while assembly of Solexa-sized simulated reads was analyzed in Warren et al. (2007), the assembly of real Solexa reads proved to be more challenging. Since Solexa reads have very few indels, we have chosen to use a heuristic error correction similar to Pevzner et al. (2001) instead of spectral alignment (SA) that performed well in the case of indel-prone 454 reads. This heuristic iteratively finds the mutation in a read that makes the most l-tuples solid (see Methods) until all tuples in the read are solid, or no mutation is found to make an l-tuple solid (we use $l = 15$). The error correction partitions the reads into 592 K of error-corrected reads and

[7]As described above, EULER-SR does not try to estimate the multiplicities of tandem repeats and misses three copies of tandem repeats in *E. coli* (approximately 1600, 1000, and 300 nucleotides) and one copy of a tandem repeat in *S. pneumoniae* (~500 nt).

402 K of nonfixable reads. The significantly higher proportion of nonfixable reads (as compared with 454 platform) reflects the higher error rates of Solexa reads and some surprising run-dependent biases in the distribution of errors along the reads. Out of 598 K error-corrected reads, 580 K are error free, 97% of all error-corrected reads (as compared with >99% in case of 454 reads). The assembly results in 82 long contigs (larger than 500 bp) of total length 100,360. The optimal assembly (as revealed by the repeat graph on 24-mers) results in 71 long contigs of total length 124,660.

Combining short read assembly with mate-pair data

One of the goals of HTSR sequencing is to sequence reads without the expensive and slow step of producing a clone library. While it may be difficult to produce paired reads in the traditional sense of sequencing opposite directions of an amplified clone, it is possible to amplify and sequence short paired-end tags in parallel. To construct the paired tags, short genomic sequences are circularized using a known linker sequence and are cleaved outside the ligation site using Type IIs restriction enzymes leaving 25–31 base tags of genomic sequence available for sequencing (Shendure et al. 2005). Although other technologies are also capable of producing longer paired ends (http://www.454.com/downloads/protocols/1_paired_end.pdf), there are currently no publicly available mate-paired short read data sets for these genomes. We therefore decided to simulate the mate-pair information to investigate how it improves the assembly.

We modified the mate-pair assembler, Euler-DB (Pevzner and Tang 2001), to run on the volume of data generated in HTSR sequencing and tested it on simulated paired-end data of short reads. The reads for *E. coli* and *S. pneumoniae* were mapped to the reference genomes, and tag libraries of sizes 2.5 kb, and 10 kb were generated by selecting 50-bp tags from the reads separated by 2500 ± 250 and 10,000 ± 1000 bases, for each library. The number of tags generated (300,000) is less than the total sequencing capacity of a typical HTSR sequencing run. The results of our assembly using mate-pairs are listed in Table 3.[8]

The current practice in most sequencing centers is to use low-coverage Sanger sequencing to complement short reads in hierarchical assembly (Goldberg et al. 2006). Because EULER-SR may be used with any length read, we simulated paired-end Sanger sequences for *S. pneumoniae* and performed assembly by combined these reads with *S. pneumoniae* 454 reads. The results are shown in Figure 1. While adding long reads results in substantial improvement in the assembly quality, we found that remarkably small number of long reads contribute to this improvement. For example, in the case of 5× read coverage by long reads, ≈96% of all mate-pairs mapped to a single contig composed of 454 reads and thus do not provide complementary information for 454-based data except for marginal potential improvement in the consensus quality. Moreover, most mate-pairs mapped to multiple contigs span only two edges in the de Bruijn graph and thus also do not improve the assembly of 454 reads. As a result, only 238 out of ≈6800 long mate-paired reads (only ≈3%) map to more than two edges in the de Bruijn graph[9] and thus meaningfully contribute to improving the assembly.

Table 3. Summary of the assemblies using real 454 reads combined with simulated mate-pairs

Genome	Assembler	No. long contigs	Total length of long contigs (in kb)	N50 (in bases)
S. pneumoniae	2.5 kb	156	2111	43,418
	10 kb	139	2224	36,253
E. coli	2.5 kb	163	5016	91,252
	10 kb	126	4897	101,647

Each end of a clone is 50 bases.

[8]The surprising deterioration of N50 statistics for 10-kb spacing (as comparing with 2.5-kb spacing) reflects ambiguities in mapping longer paths between mate-pairs in highly tangled de Bruijn graphs.

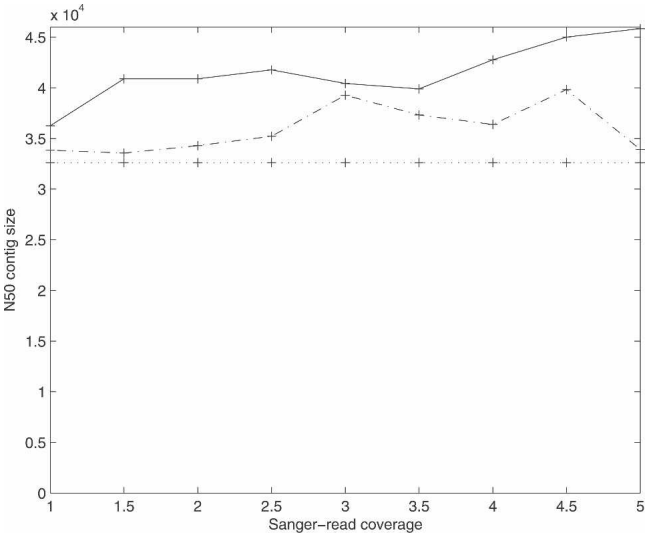


Figure 1. Assembly of *S. pneumoniae* using 454 reads complemented with simulated Sanger reads (3000 ± 300 bp spacing). Uniformly distributed 800 base reads were generated for coverage varying from 1× to 5×, and then assembled using EULER-SR. We show the assembly using mate-pairs (solid), assembly using unpaired Sanger reads (broken), and the base assembly quality of 454 reads (dotted).

Our analysis of the hybrid protocol revealed that the overwhelming majority of long reads do not improve upon the assembly already generated from short reads. It does not necessarily mean that the coverage by long reads in hybrid approaches may be reduced without affecting the assembly quality: For example, Figure 1 does not reflect the contribution of long reads to improving scaffolds. However, Figure 1 suggests that a critical review of the benefits of hybrid approaches may be necessary to compare the relative benefits and costs of generating shotgun long reads as with those of other finishing approaches.

Discussion

We presented the EULER-SR short read assembler that significantly reduces the memory requirements of previously described Eulerian assemblers and has a potential to produce nearly optimal assemblies of bacterial genomes using modest computational resources. Our assembly software is available at http://euler-assembler.ucsd.edu.

We have used assemblies produced by the proprietary Newbler assembler for benchmarks, as there are currently no other

[9]123, 72, 31, 9, and 3 mate-pairs map to 3, 4, 5, 6, and more than 6 edges correspondingly.

methods available for assembling short reads. Newbler is an excellent tool for assembling 454 reads as it uses raw flow values to mitigate sequencing errors. Our assembler compares well with Newbler yet does not rely on proprietary information, allowing it to assemble reads from multiple sequencing platforms.

Sundquist et al. (2007) recently proposed a new SHRAP experimental protocol that may enable the assembly of mammalian genomes with short reads. SHRAP uses the older version of EULER (Pevzner et al. 2004) as an engine for assembly of BAC-sized genome segments, which are then joined. One benefit of EULER-SR assembly is that assemblies of large genomes may be performed because de Bruijn graph construction is intrinsically parallelizable. The de Bruijn graph of the union of two sets of reads is simply the union of the de Bruijn graph for each independent set of reads. Therefore, it may be possible to generate short read assemblies on large genomes by simply taking the union of small sets of reads succinctly represented as condensed de Bruijn graphs.

Methods

Error correction in reads

Our assembly proceeds in several steps: error correction, graph construction, graph correction, and assembly by transforming paths in the corrected graph into contigs. Base miscalls and indels are inevitable in sequencing projects, and it is necessary to detect errors in reads to accurately determine the finished sequence. In the past this was done after assembly by mapping reads to the consensus sequence. Pevzner et al. (2001) introduced error correction before the assembly and demonstrated that it greatly simplifies the assembly. Now this is done as a standard preprocessing step before assembly using a multiple alignment of reads (Tammi et al. 2003), or by clustering and aligning reads prior to assembly (Batzoglou et al. 2002). In HTSR sequencing, constructing multiple alignments of short reads is very time consuming, and so we correct errors in reads prior to assembly using a method called spectral alignment (SA) that does not use a multiple sequence alignment. This method takes a read r and a set of l-tuples $\mathcal{S}$, called a spectrum, and finds the minimum number of substitutions, insertions, and deletions in r required to make every l-tuple in r belong to $\mathcal{S}$. The set $\mathcal{S}$ is chosen by counting the frequency of all l-tuples present in all reads in a sequencing project, and selecting tuples that occur with multiplicity above some threshold m (called solid l-tuples). An iterative solution to SA was described in Pevzner et al. (2001), followed by a dynamic programming solution in Chaisson et al. (2004). The dynamic programming solution to SA allows for efficient searching through insertions and deletions, and so it is particularly well suited for fixing errors in reads generated by pyrosequencing, in which the errors are biased toward indels. The SA approach implemented in EULER-SR is faster and has a higher rate of error correction than the approach described in Chaisson et al. (2004).

The choice of appropriate parameters for fixing errors using SA should minimize the number of erroneous l-tuples remaining in reads while not affecting correct but low-coverage sequences. Consider a sequencing project that produces N reads of average length L over a genome of length G. The average number of reads covering an l-tuple is $a = N \times (L - l)/G$ and is approximately distributed by a Poisson with parameter a. To minimize the number of correct sequences considered to be erroneous, we pick a multiplicity m for which there are few expected l-tuples covered by less than m reads. The probability an l-tuple in a genome is covered m times is

$$\sum_{y=0}^{m} P_Y(y),$$

where $Y \sim$ Poisson with parameter a. For a typical high-throughput sequencing project of a bacterial genome of 4 million bases with 1 million 100-bp reads, there are under one hundred 20-mers expected to be covered less than five times. Furthermore, under the gross simplification that the sequences are random and there is a 1% random error rate, no erroneous l-tuples are expected to be made solid by coincident errors.

de Bruijn graph construction

Because of the massive scale of the data in short read assembly, it is necessary to perform assembly in linear or close to linear time. Although it is possible to construct a de Bruijn graph in linear time, memory efficiency is more of a premium for short reads assembly than run time. Our de Bruijn graph construction is implemented in several stages, in some of which we trade linear time for methods that run on sorted lists [thus $O(n \cdot logn)$ time] but reduce memory requirements.

For a genome of length L, the de Bruijn graph has $O(L)$ vertices and $O(L)$ edges, regardless of the number of reads in the data set. We find the set of vertices in the de Bruijn graph using an efficient hashing structure in linear time proportional to the number of reads R in the data set. We then represent the vertices as a sorted list V of tuples, allowing us to discard the memory overhead of the hashing structure required for O(1) time access. We next generate adjacencies by querying the vertex list V with a binary search for every adjacent pair of l-tuples in the set of reads R. By representing the graph as an adjacency list, our de Bruijn graph uses a maximum of $O(L) * (k + l)$ bytes, where k is the memory allocated for each vertex (40 bytes in the current implementation). For subsequent phases we form a condensed de Bruijn graph, where every simple path consisting of vertices of in-degree and out-degree 1 is substituted by a single (labeled) edge. Because the reads were used to generate the de Bruijn (and thus the condensed) graph, every $(l + 1)$-tuple in a read maps to a unique edge and position in the condensed de Bruijn graph. We use this to define a path of edges for each read, and to assign a weight to every edge equal to the number of reads mapped to the edge. These paths are used after graph correction to find the Eulerian path through the de Bruijn graph that corresponds to the assembled genome.

de Bruijn graph correction

If reads covered every $(l + 1)$-tuple in the genome and were error-free, the generated de Bruijn graph G would represent the repeat graph of the genome with minimal repeat length $l + 1$. However, after error correction, a small number of errors remain resulting in some added and removed edges as compared with the repeat graph. For example, a single mutation in a read will create $(l + 1)$ extra edges in the de Bruijn graph. Our goal is to construct the graph G^* given G. If we assume every $(l + 1)$-tuple in the original genome is sequenced correctly by at least one read, then G^* is a subset of the vertices and edges of the graph on real reads G. Thus, we may perform additional error correction by detecting and removing erroneous edges on the graph G.

We adapt the methods for graph simplification on A-Bruijn graphs from Pevzner et al. (2004) to perform graph correction on de Bruijn graphs. We distinguish the two methods because graph simplification aims to capture the repeat-consensus and mosaic structure of a genome, while the goal of graph correction is to simplify the de Bruijn graph until all erroneous edges are removed, and no further. For example, in graph simplification,

repeats with high similarity are typically merged into a single edge corresponding to the repeat consensus sequence. A corrected de Bruijn graph contains a separate path for each distinct repeat sequence.

When an *A*-Bruijn graph is constructed from a set of pairwise alignments, gaps and substitutions in an alignment create undirected cycles called "bulges", and inconsistencies in an alignment create directed cycles called "whirls" (Pevzner et al. 2004). A cycle is called "short" if its girth is less than a fixed parameter g. The algorithm to generate an *A*-Bruijn graph employs a heuristic to find a graph that represents the largest possible set of consistent alignments through solving the maximum subgraph with large girth (MSLG) problem (Pevzner et al. 2004). Because this problem is hard for arbitrary girth values, a heuristic is used that replaces the *A*-Bruijn graph by its maximum spanning tree, and successively adds edges from the original graph to the spanning tree if they do not create a cycle of girth less than g.

A similar approach for removing short cycles is used for fragment assembly on the de Bruijn graphs. One property of a de Bruijn graph is that every vertex should be reachable by a directed path from the source vertex, if only one source vertex exists. A shortfall of the bulge removal method in *A*-Bruijn graphs is that when a de Bruijn graph is replaced by its undirected maximum spanning tree, certain vertices may not be reachable from the source by directed paths. Although a path straightening heuristic (Pevzner et al. 2004) is used in the *A*-Bruijn graph construction to repair edges to unreachable vertices, we circumvent this problem entirely by replacing the de Bruijn graph with its Maximum branching (Chu and Liu 1965; Edmonds 1967), using the method described in Georgiadis (2003) (the branchings have directed paths from the source vertex to all vertices). Because "alignments" are found at each vertex by perfectly matching sequences of length l, directed cycles are rarely erroneous other than those covering long low-complexity regions, for example, homopolymeric and dinucleotides. When checking to see whether adding an edge to the maximum branching forms a cycle, we distinguish between directed and undirected cycles and add an edge to the branching if it does not create an undirected cycle of girth u nor a directed cycle of length d.

While tandem repeats can be detected as whirls in the de Bruijn graph (typically directed cycles with two edges as shown in Figure 2, top), finding the number of copies (multiplicity) in a tandem repeat is a difficult problem for any assembler. For example, the multiplicity of perfect tandem repeats of length longer than half of the read length cannot be inferred from de Bruijn graphs. As a result, inferring the multiplicities of tandem repeats usually amounts to error-prone coverage analysis (e.g., a tandem repeat with multiplicity 3 is expected to have 50% more coverage than a repeat with multiplicity 2). To avoid potential errors caused by the coverage analysis (short read technologies often produce uneven coverage) and to reduce the fragmentation due to tandem repeats, we assume there are only two copies of the perfect tandem repeat and construct the assembly as a path that traverses the repeat twice, as shown in Figure 2 (bottom). While this procedure may underestimate the copy number of tandem repeats we found that it leads to very few errors.[10] We therefore believe that it is a practical approach, at least until more information about coverage and specifics of errors of short read technologies becomes available.

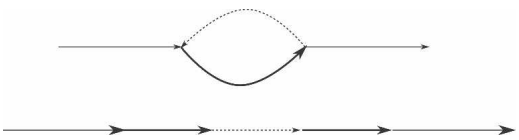


Figure 2. A tandem repeat as a whirl consisting of two edges in the condensed de Bruijn graph, and its transformation into a tandem duplication with multiplicity 2. The bold edge is the repeat, and the broken edge represents the sequence between the consecutive copies of the repeat.

Reads that are mapped to edges that have been deleted during graph correction are threaded through the remaining edges. Let the path of such a read be $(e_0, \ldots, e_i, \ldots, e_m)$, and e_i be the removed edge. We search for an alternative path $P = (a_0 \ldots a_n)$ from the start to the end of edge e_i such that the sequence corresponding to e_i and P are sufficiently similar, and replace e_i with P if such a path is found.

In addition to bulge and whirl removal, we also apply two additional techniques for removing erroneous edges: erosion as described in Pevzner et al. (2004) and low-weight edge removal. Erosion removes short sources and sinks from the graph. Low-weight edge removal is typically a way to detect low-quality chimeric reads. Although most HTSR sequencers do not use clone libraries, sequencing errors at the ends of reads create erroneous edges similar to those created by chimeric reads.

After the de Bruijn graph has been corrected we apply the equivalent transformation described in Pevzner et al. (2001) to resolve repeats that are shorter than the length of a read. If no mate-pair information exists, we cut paths that do not cover repeats with an operation similar to the x-cut (Pevzner et al. 2001). When mate-pair information is available, we postpone the cutting operation, and apply equivalent transformation with mate-pairs (Pevzner and Tang 2001).

Acknowledgments

We are indebted to Haixu Tang for much advice that was crucial for successful transformation of EULER+ into EULER-SR. We thank Xiaohua Huang and Eric Roller for many helpful discussions on emerging short read technologies. We thank Michael Egholm and James Knight for providing us with 454 reads and assemblies and helpful discussions about pyrosequencing reads. We thank Joe Ecker and Ronan O'Malley for providing us with Solexa reads and helpful discussions on BAC resequencing using Solexa platform. Finally, we are grateful to Alla Lapidus and Jeong-Hyeon Choi for advice on fragment assembly. This research is supported by NIH grant 1R21HG004130-01 and by NSF Infrastructure grant EIA-0303622.

References

Andries, K., Verhasselt, P., Guillemont, J., Gohlmann, H.W., Neefs, J.M., Winkler, H., Van Gestel, J., Timmerman, P., Zhu, M., Lee, E., et al. 2005. A diarylquinoline drug active on the ATP synthase of *Mycobacterium tuberculosis*. *Science* **307:** 223–227.

Bandeira, N., Tsur, D., Frank, A., and Pevzner, P.A. 2007. Abstract protein identification by spectral networks analysis. *Proc. Natl. Acad. Sci.* **104:** 6140–6145.

Barski, A., Cuddapah, S., Cui, K., Roh, T.Y., Schones, D.E., Wei, G., Chepelev, I., and Zhao, K. 2007. High-resolution profiling of histone methylations in the human genome. *Cell* **129:** 823–837.

Batzoglou, S., Jaffe, D.B., Stanley, K., Butler, J., Gnerre, S., Mauceli, E., Berger, B., Mesirov, J.P., and Lander, E.S. 2002. ARACHNE: A whole-genome shotgun assembler. *Genome Res.* **12:** 177–189.

Chaisson, M., Tang, H., and Pevzner, P.A. 2004. Fragment assembly with short reads. *Bioinformatics* **20:** 2067–2074.

Chu, Y.J. and Liu, T.H. 1965. On the shortest arborescence of a directed

[10] Also in cases where coverage across the tandem repeat is low, tandem repeats may be collapsed into a single copy.

graph. *Sci. Sin.* **14:** 1396–1400.
Cormen, T.H., Leiserson, C.E., and Rivest, R.L. 1995. *Introduction to algorithms*. MIT Press, Cambridge, MA.
Edmonds, J. 1967. Optimum branchings. *J. Res. Natl. Bur. Stand.* **61B:** 233–240.
The ENCODE Project Consortium. 2007. Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. *Nature* **447:** 799–816.
Feuk, L., Carson, A.R., and Scherer, S.W. 2006. Structural varaition in the human genome. *Nat. Rev. Genet.* **7:** 85–97.
Georgiadis, L. 2003. Arborescence optimization problems solvable by Edmonds' algorithm. *Theor. Comput. Sci.* **301:** 427–437.
Goldberg, S.M., Johnson, J., Busam, D., Feldblyum, T., Ferriera, S., Friedman, R., Halpern, A., Khouri, H., Kravitz, S.A., Lauro, F.M., et al. 2006. A Sanger/Pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes. *Proc. Natl. Acad. Sci.* **103**: 11240–11245.
Huang, X., Wang, J., Aluru, S., Yang, S., and Hillier, L. 2003. PCAP: A whole-genome assembly program. *Genome Res.* **13:** 2164–2170.
Idury, R.M. and Waterman, M.S. 1995. A new algorithm for DNA sequence assembly. *J. Comput. Biol.* **2:** 291–306.
Jaffe, D.B., Butler, J., Gnerre, S., Mauceli, E., Lindblad-Toh, K., Mesirov, J.P., Zody, M.C., and Lander, E.S. 2003. Whole-genome sequence assembly for mammalian genomes: Arachne 2. *Genome Res.* **13:** 91–96.
Jiang, Z., Tang, H., Ventura, M., Cardone, M.F., Marques-Bonet, X., She, X., Pevzner, P.A., and Eichler, E.E. 2007. Ancestral reconstruction of segmental duplications reveals punctuated cores of human genome evolution. *Nat. Genet.* **39:** 1361–1368.
Kim, J.B., Porreca, G.J., Song, L., Gorham, J.M., Church, G.M., Seidman, C.E., and Seidman, J.G. 2007. Polony multiplex analysis of gene expression (PMAGE) in mouse hypertrophic cardiomyopathy. *Science* **316:** 1481–1484.
Lippert, R.A., Mobarry, C.M., and Walenz, B.P. 2005. A space-efficient construction of the Burrows-Wheeler transform for genomic data. *J. Comput. Biol.* **12:** 943–951.
Margulies, M., Egholm, M., Altman, W.E., Attiya, S., Bader, J.S., Bemben, L.A., Berka, J., Braverman, M.S., Chen, Y.J., Chen, Z., et al. 2006. Genome sequencing in microfabricated high-density picolitre reactors. *Nature* **437:** 376–380.
Medvedev, P., Georgiou, K., Myers, G., and Brudno, M. 2007. Computability of models for sequence assembly. In *Lecture notes in computer science,* Vol. 4645, pp. 289–301. Springer, Heidelberg.
Metzker, M.L. 2005. Emerging technologies in DNA sequencing. *Genome Res.* **15:** 1767–1776.
Myers, E.W. 2005. The fragment assembly string graph. *Bioinformatics* (Suppl. 2) **21:** ii79–ii85. doi: 10.1093/bioinformatics/bti1114.
Pevzner, P.A. and Tang, H. 2001. Fragment assembly with double-barreled data. *Bioinformatics* (Suppl. 1) **17:** S225–S233.
Pevzner, P.A., Borodovsky, M.Y., and Mironov, A.A. 1989. Abstract linguistics of nucleotide sequences. ii: Stationary words in genetic texts and the zonal structure of DNA. *J. Biomol. Struct. Dyn.* **6:** 1027–1038.
Pevzner, P.A., Tang, H., and Waterman, M.S. 2001. An Eulerian path approach to DNA fragment assembly. *Proc. Natl. Acad. Sci.* **98:** 9748–9753.
Pevzner, P.A., Tang, H., and Tesler, G. 2004. De novo repeat classification and fragment assembly. *Genome Res.* **14:** 1786–1796.
Raphael, B.J., Zhi, D., Tang, H., and Pevzner, P.A. 2004. A novel method for multiple alignment of sequences with repeated and shuffled domains. *Genome Res.* **14:** 2336–2346.
Ronaghi, M., Mathias, U., and Nyren, P. 1998. DNA sequencing: A sequencing method based on real-time pyrophosphate. *Science* **281:** 363–365.
Shendure, J., Porreca, G.J., Reppas, N.B., Lin, X., McCutcheon, J.P., Rosenbaum, A.M., Wang, M.D., Zhang, K., Mitra, R.D., and Church, G.M. 2005. Accurate multiplex polony sequencing of an evolved bacterial genome. *Science* **309:** 1728–1732.
Sundquist, A., Ronaghi, M., Tang, H., Pevzner, P., and Batzoglou, S. 2007. Whole-genome sequencing and assembly with high throughput short read technologies. *PLoS One* **2:** e484. doi: 10.1371/journal.pone.0000484.
Tammi, M.T., Arner, E., Kindlund, E., and Andersson, B. 2003. Correcting errors in shotgun sequences. *Nucl. Acids Res.* **31:** 4663–4672. doi: 10.1093/nar/gkg653.
Warren, R.L., Sutton, G.G., Jones, S.J., and Holt, R.A. 2007. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics* **23:** 500–501. doi: 10.1093/bioinformatics/btl629.
Whiteford, N., Haslam, N., Weber, G., Prügel-Bennett, A., Essex, J.W., Roach, P.L., Bradley, M., and Neylon, C. 2005. An analysis of the feasibility of short read sequencing. *Nucleic Acids Res.* **33:** e171. doi: 10.1093/nar/gni170.
Zhi, D., Raphael, B., Price, A., Tang, H., and Pevzner, P.A. 2006. Identifying repeat domains in large genomes. *Genome Biol.* **7:** R7.

Received August 28, 2007; accepted in revised form October 16, 2007.