BRIEFINGS IN BIOINFORMATICS. VOL 11. NO 5. 457–472
Advance Access published on 19 August 2010

doi:10.1093/bib/bbq020

De novo assembly of short sequence reads

Konrad Paszkiewicz and David J. Studholme

Submitted: 3rd March 2010; Received (in revised form): 7th May 2010

Abstract

A new generation of sequencing technologies is revolutionizing molecular biology. Illumina's Solexa and Applied Biosystems' SOLiD generate gigabases of nucleotide sequence per week. However, a perceived limitation of these ultra-high-throughput technologies is their short read-lengths. *De novo* assembly of sequence reads generated by classical Sanger capillary sequencing is a mature field of research. Unfortunately, the existing sequence assembly programs were not effective for short sequence reads generated by Illumina and SOLiD platforms. Early studies suggested that, in principle, sequence reads as short as 20–30 nucleotides could be used to generate useful assemblies of both prokaryotic and eukaryotic genome sequences, albeit containing many gaps. The early feasibility studies and proofs of principle inspired several bioinformatics research groups to implement new algorithms as freely available software tools specifically aimed at assembling reads of 30–50 nucleotides in length. This has led to the generation of several draft genome sequences based exclusively on short sequence Illumina sequence reads, recently culminating in the assembly of the 2.25-Gb genome of the giant panda from Illumina sequence reads with an average length of just 52 nucleotides. As well as reviewing recent developments in the field, we discuss some practical aspects such as data filtering and submission of assembly data to public repositories.

Keywords: *sequence assembly; Illumina; SOLiD; genome*

INTRODUCTION

A new generation of sequencing technologies is revolutionizing molecular biology [1–15]. They have dramatically lowered the costs per sequenced nucleotide and increased throughput by orders of magnitude. Illumina's Solexa and Applied Biosystems' SOLiD can generate gigabases of nucleotide sequence per week. However, a perceived limitation of these ultra-high-throughput technologies is their short read-lengths. The first incarnation of Illumina's Genome Analyzer (GA) typically generated sequence reads of just 36-nucleotides long. Currently Illumina support generation of up to 100-nucleotide read-lengths, but many laboratories are still opting for read-lengths shorter than this and currently SOLiD generates reads of 30–50 nucleotides. Until recently, it was generally assumed that such short read-lengths would restrict these technologies to resequencing applications. Resequencing [12, 13] implies the availability of a reference sequence (usually a complete genome sequence) against which the short reads can be aligned. The objective of resequencing is the discovery of single-nucleotide polymorphisms (SNPs) or other genetic variations between individuals of the same species or between healthy and cancerous cells. In the absence of a reference sequence against which to compare sequence reads, the key step in computational analysis is *de novo* sequence assembly. Several recently published reviews have dealt with the assembly of short sequence reads [15–20]. The current manuscript complements these by focussing on practical issues facing the biologist who wants to leverage the low cost and high-throughput of short-read sequence technologies for *de novo* assembly. We mainly focus on Illumina's Solexa technology [21], because this is

Corresponding author. David J. Studholme, School of Biosciences, Geoffrey Pope Building, University of Exeter, Stocker Road, Exeter, EX4 4QD, UK. Tel: +44 (0) 1392 724678; Fax: +44 (0) 1392 263434; E-mail: d.j.studholme@exeter.ac.uk
Konrad Paszkiewicz trained as a physicist before his PhD in structural bioinformatics at Imperial College, London. He then carried out post-doctoral research at Peninsula Medical School before being appointed as a Senior Experimental Officer at Exeter University in 2008 where he manages the in-house sequencing facility and runs the Bioinformatics support service (ExBiSS).
David J. Studholme joined Exeter University in November 2009 as a Senior Lecturer. He previously led the bioinformatics team at The Sainsbury Laboratory in Norwich, UK.

the platform with which we are most familiar and that appears most frequently in the published primary literature. However, much of what we write applies equally to assembly of sequence reads generated by Applied Biosytems' SOLiD [22] and Complete Genomics [23].

THE CHALLENGE OF ASSEMBLING SHORT SEQUENCE READS

De novo sequence assembly is the process whereby we merge together individual sequence reads to form long contiguous sequences ('contigs') sharing the same nucleotide sequence as the original template DNA from which the sequence reads were derived. Assembly of sequence reads generated by classical Sanger capillary sequencing is a mature field of research [17]. Two main types of algorithm are commonly used: (i) the overlap-layout-consensus (OLC) approach and (ii) algorithms based on a de Bruijn graph. These have been well reviewed previously [15–20] and have been implemented in very effective genome-assembly software packages including Arachne [24], AMOS [25], Atlas [26], Celera Assembler [27], CAP3 [28], Euler [29], PCAP [30], Phrap [31], RePS [32] and Phusion [33]. Unfortunately, these genome sequence assembly programs are not well suited to short sequence reads generated by Illumina and SOLiD platforms, for several reasons. First, the numbers of reads produced by Illumina and SOLiD systems are around three to four orders of magnitude greater per run per instrument per day than that from capillary sequencers. OLC methods require overlaps to be scored between all possible pairs of reads, which is computationally costly, scaling with the square of the number of reads. Second, the shorter sequences mean any base-calling errors have a much greater potential impact. Third, unique overlaps between pairs of reads are much less likely, given the short read-lengths. Finally, it is often not possible to resolve repetitive sequences where the length of the repeated unit is longer than the sequence read. Given these challenges, most recent *de novo* genome sequencing projects have opted for Roche's 454 GS-FLX sequencing technology (rather than the Illumina or the SOLiD platforms). These decisions were probably influenced by its significantly lower cost per nucleotide (compared to capillary sequencing) yet longer read-length (compared to Illumina and SOLiD) and the availability of Roche's proprietary 'Newbler' assembler, which is optimized for GS-FLX data. Another approach has been to perform 'hybrid' assemblies of mixtures of both long and short reads in an attempt to gain the respective benefits of both (e.g. [34–38]). However, given the even lower per-nucleotide costs of Illumina and SOLiD, some researchers, especially those with access to a local instrument, have been tempted to use these short-read platforms for *de novo* genome sequencing. This has led to the generation of several draft genome sequences (Table 1) [39–48] based exclusively on short sequence Illumina sequence reads, recently culminating in the assembly of the 2.25 Gb genome of the giant panda from Illumina sequence reads with an average length of just 52 nucleotides [40, 49]. Of the currently published genome assemblies built exclusively from short reads, most are no more than a few megabases in length and they still contain on the order of hundreds of gaps. The assembly of the panda genome from short reads, although it is a momentous achievement, contains significantly more gaps than previous mammalian genome assemblies based on longer reads and questions have been raised about its completeness and accuracy [49].

Table 1: Genomes assembled *de novo* exclusively from Illumina short sequence reads

Organisms	References
Turkey (*Meleagris gallopavo*)	[39]
Giant panda (*Ailuropoda melanoleuca*)	[40]
Bacillus subtilis 168	[57]
Bacillus subtilis natto	[41]
Pseudomonas syringae pv. tabaci 11528	[42]
Pseudomonas syringae pv. syringae Psy642	[43]
Pseudomonas syringae pv. tomato T1	[44]
Pseudomonas syringae pv. aesculi	[45]
Apple scab (*Ventura inaequalis*)	[46]
Pine (*Pinus* species) chloroplast	[47, 48]

THE FEASIBILITY OF ASSEMBLING GENOMES EXCLUSIVELY FROM SHORT SEQUENCE READS

Early studies [50, 51] suggested that, in principle, sequence reads as short as 20–30 nucleotides could be used to generate useful assemblies of both prokaryotic and eukaryotic genome sequences, albeit containing many gaps. For example, using reads of 30 nucleotides, 75% of the prokaryotic *Escherichia coli* genome could be assembled into contigs of longer than 10 kb and 96% of genes were covered by a single contig [50]. For assemblies of the larger and more repetitive genome of the eukaryote *Caenorhabditis elegans*, 51% of the genome is covered by contigs of at least 10 kb. More recently, Kingsford and colleagues [52] examined the complete nucleotide sequences of 408 prokaryotic chromosomes and calculated the best possible assemblies achievable from idealized data, that is error-free reads uniformly and comprehensively distributed over the genome. Consistent with the earlier study [50], they found that from 25-nucleotide reads it was possible to assemble at least 90% of the genes from 89% of the chromosomes. Not surprisingly, they found that longer reads tended to yield more contiguous assemblies. The biggest improvements came with increases from 25 to 35 nucleotides; beyond 35 nucleotides, increasing read-length yielded diminishing returns. Interestingly, they found that contiguity of assembly did not always correlate closely with genome size. For example, the genome of *Yersinia pestis* yielded much smaller contigs than that of *E. coli*, even though both genomes are around 4.4 Mb. In practice, this means that there is no easy formula for predicting the quality of an assembly as a simple function of genome size. It also means that we cannot predict the quality of a genome assembly for one species based on results from another species. On a more optimistic note, this study suggested that for most prokaryotic genomes it is theoretically possible to assemble the vast majority of protein-encoding genes from reads of no longer than 40 nucleotides. Those genes that failed to assemble correctly were mostly associated with repetitive elements such as transposons.

REAL DATA YIELD POORER ASSEMBLIES THAN DO SIMULATED IDEALIZED DATA

These feasibility studies demonstrated the potential for generating informative draft genome assemblies entirely from short sequence reads; but there were several caveats. Although a significant proportion of the genome might be represented in large contigs, the assembly contained many gaps and large numbers of short contigs. Therefore, the precise order of the contigs is unknown, limiting studies of synteny and genome rearrangements. Even studies aimed at gene-discovery could not be exhaustive; at least 2% of the genes were disrupted in the average prokaryotic genome assembly from 50-nucleotide reads [52]. Furthermore, this study presents a best-case scenario; that is, it describes the upper limits of what is achievable, based on idealized simulated data. The errors and bias observed in real sequence data can dramatically reduce the quality of assemblies. Real Illumina sequence data are different from idealized data in two important respects: real sequence reads contain errors and the distribution of sequence reads is not uniform (nor random) over the genome (Figures 1 and 2). Both of these phenomena have a negative effect on sequence assembly. This is illustrated empirically by comparing the assembly (using Velvet) of a simulated error-free dataset of 36-nucleotide reads (from *Pseudomonas syringae* pathovar *syringae* B728a) against an equivalent Illumina dataset from the same strain [53]. The simulated dataset yielded longer contigs ($N_{50} = 13\,771$ versus 6963 nucleotides) that contained significantly fewer errors. We also found that real Illumina data from the *E. coli* genome produced poorer assemblies than simulated ideal data (Figure 3). Several previous studies have reported non-uniform and non-random representation over the genome [54–57]. This may be partly explained by amplification biases, correlating with sequence properties such as G+C content [54, 55]. In our experience, the degree of bias can vary between samples even from very similar template sequences, thus leading to unpredictable assembly quality. Furthermore, with bacterial genomes, we often observe a systematic over-representation of sequences close to the origin of replication (Figure 1), a pattern also reported by refs. [56, 57]. The effects of such biases on quality of sequence assembly have not been systematically analysed, though they will likely be detrimental. For example, the Celera assembler uses distributions of read start points to resolve repeats. Several of the most popular assembly software packages assume random sampling of the genome in order to correct sequence errors. EULER [58, 59] and ALLPATHS [60] attempt to correct errors in reads prior to assembly, whilst Velvet [61] and FragmentGluer [62] deal with errors by editing the graphs. Velvet assumes a bimodal frequency distribution for the depth of coverage of contigs. The peak at shallow coverage is assumed to comprise errors and eliminates them from the assembly. The user can set a parameter called 'coverage cut-off' to specify the stringency of this

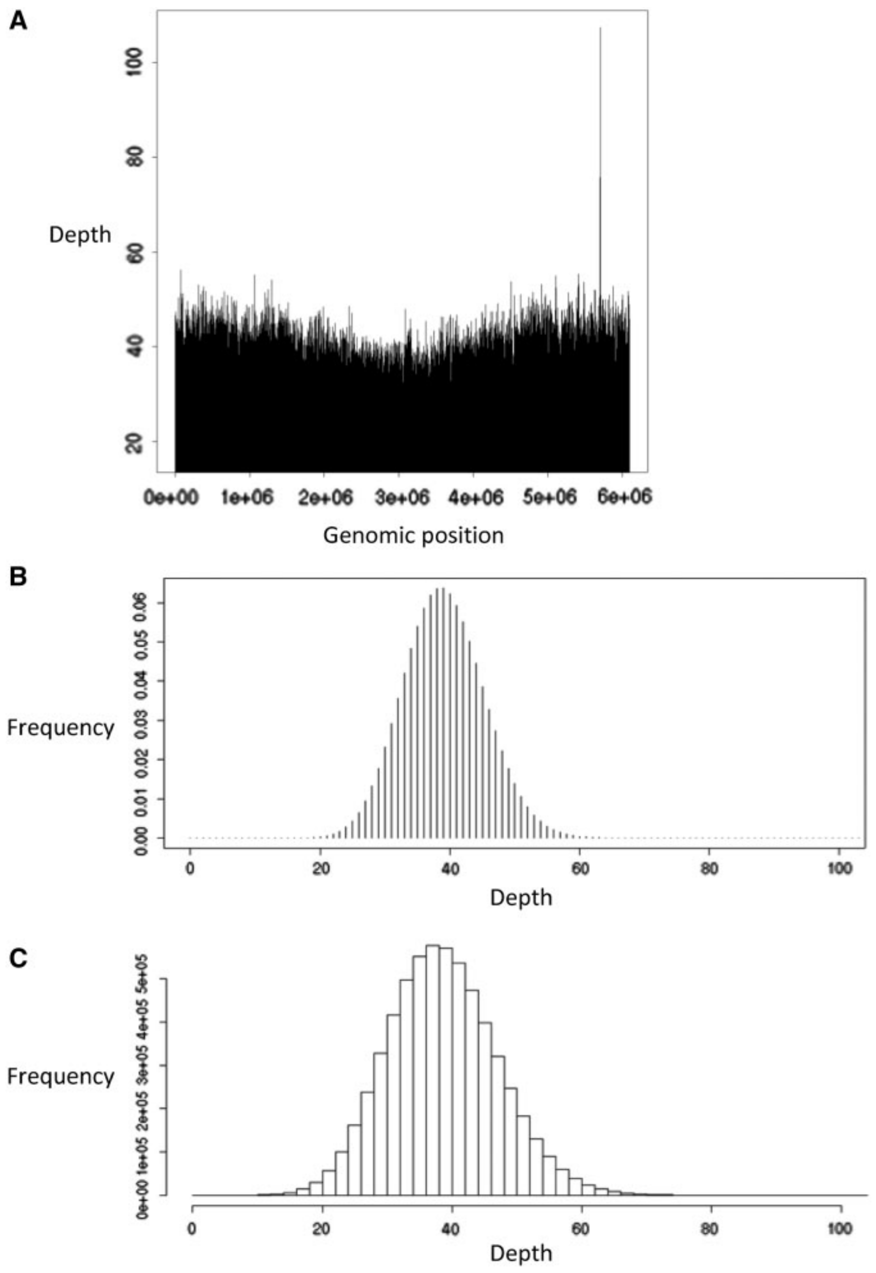

Figure 1: Biases in real short-read sequence data. A set of Illumina 36-nucleotide genomic sequence reads [53] from *P. syringae* pathovar *syringae* B728a was aligned against the previously published [116] genome sequence of this strain. (**A**) Illustrates the depth of coverage by aligned reads over the 6 Mb circular chromosome. Coverage is shallower around the 3 Mb region than it is near the origin of replication (position 0), (**B**) illustrates the expected frequency distribution of alignment depth, assuming random sampling of the genome and (**C**) illustrates the observed frequency distribution of alignment depth, which is broader than the expected distribution, indicating greater variance due to biased sampling.

purging of low-coverage contigs. Given the same total quantity of sequence data, over-representation of some genomic sequences inevitably means shallower coverage of other parts of the genome. It has been well established that assembly contiguity and accuracy are poor at very shallow coverage (e.g. [53]).

NOT ALL ASSEMBLERS ARE EQUAL—CHOOSING THE RIGHT TOOL FOR THE JOB

The early feasibility studies and proofs of principle [50, 51] inspired several bioinformatics research groups to implement new algorithms as freely

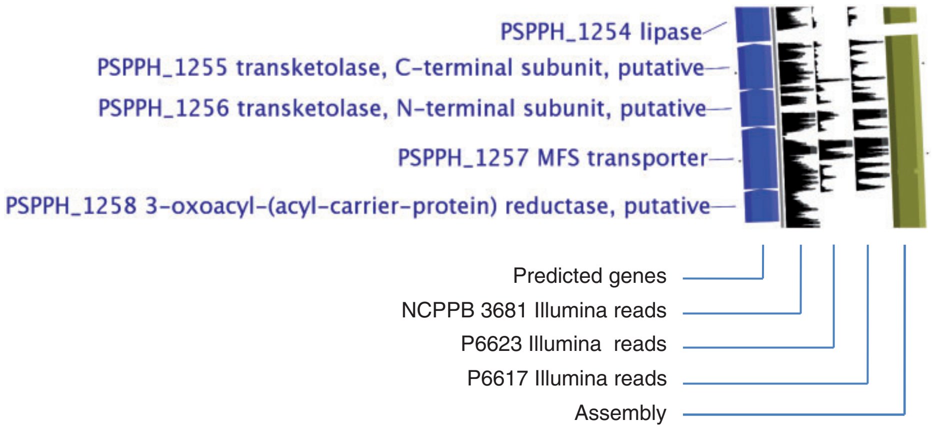
PSPPH_1254 lipase
PSPPH_1255 transketolase, C-terminal subunit, putative
PSPPH_1256 transketolase, N-terminal subunit, putative
PSPPH_1257 MFS transporter
PSPPH_1258 3-oxoacyl-(acyl-carrier-protein) reductase, putative
Predicted genes
NCPPB 3681 Illumina reads
P6623 Illumina reads
P6617 Illumina reads
Assembly

Figure 2: Biases in real short-read sequence data. Shown is a section of the *P. syringae* pathovar *phaseolicola* 1448A genome [116], comprising five predicted genes. Sets of Illumina 36-nucleotide genomic sequence reads from three strains (NCPPB 3681, P6617 and P6623) of *P. syringae* pathovar *aesculi* were aligned against the 1448A genome using MAQ [117]. The thickness of the black tracks indicates the depth of the alignment. The distribution of the sequence reads aligned on the genome is very variable both within each strain dataset and among the strains, illustrating significant bias in sampling of the genome by Illumina sequencing and variation in the bias among datasets.

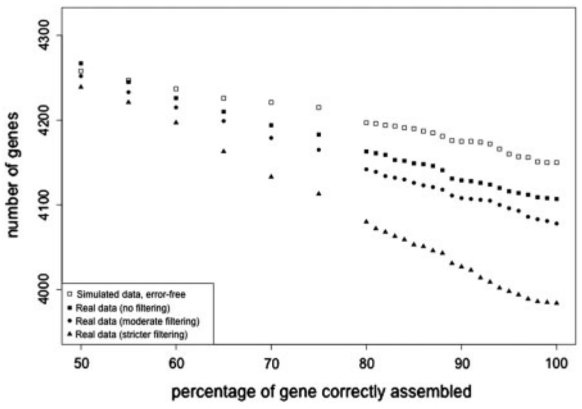

Figure 3: Real data produces poorer assemblies than simulated idealized data. We downloaded the *E. coli* strain K-12 MG1655 dataset SRR001665 from the NCBI Sequence Read Archive and assembled using Velvet 0.7.61 with VelvetOptimiser 2.1 (script distributed with the Velvet package). We also subjected the data to two quality filtering regimes, removing all reads where 90% of the nucleotides had a quality score below 15 (moderate) or below 20 (stricter). We also generated an equivalently sized set of simulated error-free sequence 36-nucleotide reads. We assayed the quality of each assembly by calculating the fraction correctly assembled for each of the 4318 genes.

available software tools specifically aimed at assembling reads of 30–50 nucleotides in length. These programs [60, 61, 63–96] are listed in Table 2. It is essential for the genome biologist to have at least a basic knowledge of the underlying algorithms in order to make rational decisions about analysing his/her data. All three types of algorithm have been extensively reviewed elsewhere (e.g. [17, 19] and so are only briefly described here. Some programs attempt to further assemble contigs into longer assemblages known as 'supercontigs' or 'scaffolds'. This step is commonly known as 'scaffolding', and exploits additional information available in paired sequence reads and/or conservation of gene-order in related biological species. Scaffolding is briefly discussed further in a following section. Although the assembly programs are all based on a small number of algorithms, they differ from each other in the details of how they deal with errors, inconsistencies and ambiguities. It should also be noted that scaffolding is not the only use for paired reads. Some assemblers, notably EULER [58], utilize read-pairing information during the assembly of contigs prior to any scaffolding step.

Assembly algorithms

The sequence assembly problem is essentially one of constructing a DNA sequence superstring that explains the observed set of sequence reads. This superstring might represent the chromosome, bacterial artificial chromosome, or other template DNA that was subjected to sequencing. If the data were completely error-free, then we would expect every sequence read to be contained within the superstring. So, we might be tempted to formulate sequence assembly as finding a superstring that

Table 2: Software for assembling short sequence reads

Program	Current version (as of 27 February 2010)	References
ABYSS	1.1.2 (15 February 2010)	[63, 64]
ALLPATHS	3 (3 December 2009)	[60, 65, 66]
CLC NGS Cell 3.0	2.2.1 (28 July 2009)	[67]
Curtain[a]	0.0.2 (16 December 2009)	[68]
Edena	2.1.1 (17 March 2008)	[69, 70]
Euler-SR	1.1.12 (30 March 2009)	[71–73]
FuzzyPath	3.1 (8 November 2009)	[74, 75]
Oases[b]	0.1.4 (29 January 2010)	[76]
QSRA	No version number (11 March 2009)	[77, 78]
SASSY	No version number	[17]
SeqCons	No version number (24 September 2009)	[79, 80]
SHARCGS	No version number (19 November 2007)	[82]
SHORTY	2.0 (date unknown)	[83, 84]
SOAPdenovo	2.20 (13 August 2009)	[85, 86]
SOPRA	1 (date unknown)	[87, 88]
SSAKE	3.4 (14 April 2009)	[89, 90]
Taipan	1.0 (15 May 2009)	[91, 92]
VCAKE	1.1 (29 June 2009)	[93, 94]
Velvet	0.7.59 (19 February 2010)	[61, 95, 96]

These software packages are able to perform *de novo* assembly of Illumina short sequence reads with the exception of SHORTY, which is designed to assemble ABI SOLiD colour-space data. Velvet and SOPRA can assemble sequence-space and colour-space data. [a]Curtain is a pipeline, based on Velvet, for hierarchical assembly of short sequence reads in order to overcome memory usage limitations. [b]Oases is specifically designed for assembling transcribed sequences.

contains all sequence-read strings as substrings. In real biological sequence data, the problem is more complicated. Sequencing error rates may be as high as 1–4% per nucleotide implying that many of the sequence reads contain mismatches with respect to the solution superstring. Furthermore, there will inevitably be multiple solutions; that is, we could propose many possible superstrings that satisfy the criterion of containing all the observed sequence reads. So which superstring is the best one? In a spirit of parsimony, we might choose the shortest superstring, but this would likely not be the biologically correct one. The reason is that real genomic DNA sequences tend to contain large numbers of perfectly and/or imperfectly repeated sequences, which would be erroneously collapsed in the shortest superstring. Some of the major differences between the popular assembly programs are to be found in their strategies for dealing with repeats. For example, whilst the Celera assembler [27] masks-out repeat sequences, EULER proactively utilizes repeats to determine contig order. A further complication is that sequence reads can originate from the reverse complement as well as from the forward orientation of the template DNA sequence.

Several assembly algorithms have been proposed and implemented as excellent software packages that can efficiently and accurately assemble capillary-based sequence reads of 400–1000 nucleotides in length into long contiguous sequences. Examples include PCAP [30], CAP3 [28], ARACHNE [24], Celera assembler [27], Phusion [33], Atlas [26] and Phrap [31]. All of these take a common approach, known as overlap-layout-consensus (OLC). The EULER assembler [58, 59] takes a different approach, based on de Bruijn graphs, which will be discussed below. The first step in OLC is finding overlaps between sequence reads. If we use the jigsaw puzzle as a metaphor for sequence assembly [58, 59], then this step is analogous to trying all possible pairs of pieces to see which ones fit together. The step of putting these pieces together corresponds to the layout step. The final step is to derive a consensus sequence from the layout.

The overlap step essentially involves an all-versus-all sequence comparison; each pair of sequence reads is ascribed a similarity score. This step is computationally expensive and scales quadratically with the number of sequence reads. However, algorithmic tricks such as ARACHNE's sort-and-extend strategy [24] can drastically improve this situation. The next step, generation of a layout from the overlaps, is often tackled using the principles of graph theory [97, 98], though some popular OLC assemblers (e.g. ARACHNE, PCAP, CAP3) do not explicitly formulate it as such. The first formal description of OLC assembly in terms of graph theory was that of ref. [97], which introduced the concept of the overlap graph. The overlap graph consists of a set of vertices (or nodes), each representing a sequence read, connected by edges representing overlaps. Finding an assembled sequence that explains the observed sequence reads becomes equivalent to finding a path through the graph that visits every vertex exactly once; this is known as a Hamiltonian circuit. Finding a Hamiltonian circuit in a graph is an NP-hard problem, which means that for all but the smallest and simplest graphs, it is effectively impossible to calculate an optimal solution. Instead, the OLC assembler must use an alternative pragmatic approach such as approximation or heuristics. In practice, OLC-based sequence assemblers such as CAP3 and

Phrap use a greedy strategy to generate a layout from the set of overlaps. This consists of progressively merging pairs of strings displaying the greatest overlap scores until only a single string remains. This approach has the advantage of being intuitively easy to understand but will not necessarily yield the globally optimal solution; indeed the final assembly sequence may depend on the order in which equally scoring pairs are merged.

The lack of an efficient algorithm for the solution of the layout problem (finding a Hamiltonian path in an overlap graph) can be circumvented by taking an alternative approach to assembly based on the de Bruijn graph [58, 59, 99, 100]. This approach also eliminates the need for an overlap phase (i.e. pairwise comparisons of all the reads). In this approach, the metaphoric jigsaw pieces (that is, the sequence reads) are first cut into even smaller pieces [58, 59]; from the set of sequence reads, a spectrum of *k*-tuples is generated. A *k*-tuple is a DNA sequence word of length *k*. For example, the sequence ATGTGCC GCA contains the three-tuples ATG, TGT, GTG, TGC, GCC, CCG, CGC and GCA. In the de Bruijn graph (or spectrum graph), the vertices (nodes) are $(k-1)$-tuples that occur in the *k*-tuples. In our example, the vertices would include two-tuples AT, TG, GT, etc. An edge that connects a pair of the vertices represents a *k*-tuples in which both $(k-1)$-tuples are present. For example, AT and TG are connected by an edge that represents the *k*-tuple ATG. The problem of assembly is now to find a shortest (or minimum weight) path or circuit that visits every edge, also known as the Chinese Postman Problem. If there exists an Eulerian path (one that passes through each edge only once), then that is the optimal solution. Finding an Eulerian path is much easier computationally than finding a Hamiltonian path. However, just as in the overlap graph, there still remains the problem of multiple solutions (due to repetitive sequences that cause circuits in the de Bruijn graph) and sequencing errors can cause de Bruijn graphs to become highly branched and tangled. The EULER assembler [58, 59] can use information from paired reads to untangle the graph. EULER [58, 59] and most of the widely used assemblers designed for short sequence reads are based on the de Bruijn graph and Eulerian path approach. They differ from each other in details of implementation and in their strategies for dealing with sequencing errors and resolution of repeats. In the toy example above, the value of *k* was 3. In practice, the bioinformatician assembling short reads is likely to choose a value of *k* between ~19 and the read-length.

Myers [97] has proposed a third alternative to the overlap graph and the de Bruijn/spectrum graph. He introduced the concept of a string graph, which is a simplified graph derived from one in which the vertices are sequence reads and the edges are overlaps, and described algorithms for efficiently generating a string graph from a set of sequence reads. The method consists of four steps: (i) an overlap graph is generated from an all-against-all alignment of the sequence reads, (ii) the overlap converted to a string graph by merging and reducing redundant overlaps and edges, (iii) false vertices and edges are identified and eliminated using a network flow approach, and (iv) an Eulerian path or circuit is found, which defines the assembly. This is much easier to find than the Hamiltonian path of an overlap graph. Thus Myers' work demonstrates that the cutting-up of reads into *k*-tuples is probably unnecessary. To our knowledge, the string graph approach has not yet been used for assembly of short sequence reads.

Methods for scaffolding

Once assemblers have generated contigs using one of the above methods, it is necessary to group contigs together in the correct orientation and order. Typically this is done by exploiting the additional information offered by paired-end reads. A read-pair that spans two contigs is taken as evidence for the juxtaposition of those two contigs within the genome. Again, this problem can be formulated using a graph-based approach; this time, the contigs are modelled as nodes (vertices) and matching read-pairs are modelled as edges connecting the pair of contigs. Again the algorithm consists of finding an optimal path through the graph. In practice, there is often inconsistency in the pattern of read-pair links arising from formation of chimaeric DNA molecules and other technical issues. In the original *Drosophila melanogaster* genome assembly project [27], 10–20% of the pairing information was believed to be false. For example, the SOAPdenovo assembler uses a minimum of three read pairs as the criterion for defining order and distance between contigs [85], whilst for ABySS the default criterion is five read-pairs [64]. Scaffolding algorithms may attempt to minimize incongruence between the proposed assembly and the observed read-pair data using consensus from large numbers of read-pairs. The EULER assembler [59] maps read pairs against the de Bruijn graph. Where a pair of reads matches two hitherto unlinked components of the graph, EULER finds a path between these reads. The estimated clone length should be approximately equal to the distance between the reads; this criterion is used to identify erroneous pairing data and to choose between alternative paths.

Any scaffolding algorithm must solve three main problems as described by ref. [19]: (i) find all connected components in the defined graph, (ii) find a consistent orientation for all nodes in the graph and (iii) given length estimates of the edges, embed the graph on a line/circle to minimize the number of constraints that are invalidated. The last two problems are defined as NP-complete, but there are good heuristics available to overcome this. Also, in the final step, it is not necessary to minimize completely as the presence of multiple edges from a single node can indicate the presence of mis-assemblies that may require manual intervention and/or additional data to correct.

Short reads make the added information offered by paired-end data essential for *de novo* assembly of all but the simplest genomes. The fundamental limitation of any read length is that any contigs resulting from an assembly will necessarily be limited to regions of the genome that do not have repetitive elements longer than the read length. Shorter read lengths make this problem particularly difficult. Additionally non-unique elements arising from gene-duplication or multi-copy domains can lead to mis-assemblies of contigs.

To resolve these problems paired-end data can be used. Essentially each section of DNA is sequenced twice—once from each end. By carefully controlling this length of DNA (typically either 200–800 nucleotides, or with a circularization protocol 1–10 kb, the so-called insert size) it is possible to obtain an estimate for how far apart each read should appear in the final assembly. In this way, assuming that at least one read can be mapped to a unique position it is possible to assign at least an approximate location for its partner. A combination of coverage deviation from the median level of coverage and paired-end information has been found to be sufficient in most cases to obtain a good assembly for even relatively long repetitive regions.

Paired reads are not the only source of long-range information useful for scaffolding. Another approach is to use long reads or contigs from another sequencing technology such as 454 of capillary sequencing. The Velvet assembler, for example, can take as input mixtures of long and short sequences. Alternatively, contigs assembled from short reads can be combined with longer sequences using a long-read assembler such as Minimus [101]. Other approaches to scaffolding include exploiting physical and/or genetic maps. A method of scaffolding based on optical maps is implemented in the SOMA software [102].

Choosing an assembler

For the biologist faced with assembling real data, which of the programs in Table 2 is the 'best'? First, we should point out that any answer to that question might soon become out of date as this is an active field and existing software is continually being improved whilst new programs are being developed. However, the key issues likely to remain are usability and assembly quality. Usability comprises a number of factors including hardware and software requirements, ease of installation and execution, and speed. The quality of an assembly comprises both the contiguity (lengths of the contigs or scaffolds) and the accuracy of the assembly. Cultural issues may also be important. For example, the community of Velvet users have reached a 'critical mass' such that there is active discussion on the Velvet mailing list [103].

Most of the published papers describing individual assembly programs include a comparison of the quality of its assembly results with those of other programs. As far as we are aware, there is not yet a comprehensive survey of assembly quality among the different programs published a by a 'neutral' research group. Such a survey would not be a straightforward undertaking, since the quality of the resulting assemblies is likely a function of both the choice of program and the particulars of the dataset; that is some programs may perform better on some datasets than on others. Different programs vary in the details of how they resolve errors and inconsistencies in the data. The nature of these errors and consistencies likely vary between haploid versus diploid genomes, for example, and further vary according to frequency of heterozygosity. An additional complication is that each assembly program takes several options and parameters. For example, in algorithms based on the de Bruijn graph, results are highly dependent on the choice of *k*-mer (i.e. *k*-tuple) size. This means that, even after having chosen which software to use, it is equally important to choose the optimal parameter values. So there is no definitive answer as to which is the 'best' short-read assembler and there is an urgent need for a comprehensive comparison (or competition) between the candidates on a suitably broad selection of datasets. Such a study should utilize a range of different datasets varying in factors such as size, error-rate, heterozygosity, repeat structure, sequence complexity, sampling bias, read lengths, insert lengths (for paired reads or mate pairs), etc. In the meantime, we would make some suggestions based on head-to-head comparisons in the literature as well as our own experience. For assembling a single library of non-hierarchical whole-genome data, with a single insert-length, from small genomes (<40 Mb), then Velvet would be as good a choice as any. Simpson and colleagues [63] found that ABySS, Velvet and EULER-SR performed much better than SSAKE and Edena on Illumina reads from the *E. coli* genome. Velvet generally runs much faster and with a smaller memory footprint than ABySS for relatively small datasets (e.g. bacterial genomes). On Illumina datasets from five microbial genomes, Velvet gave longer scaffolds and greater accuracy than EULER-SR [65, 95]. ALLPATHS2 yielded significantly more contiguous and more accurate assemblies but only when supplied with multiple DNA libraries with different insert lengths [65]. SOAPdenovo produced more contiguous and more complete assemblies of a human genome than did ABySS and also produced better assemblies than ABySS, Velvet, EULER-SR, SSAKE and Edena on *E. coli* genome [85]. QSRA yielded assemblies of chloroplast and bacterial genomes that were competitive with Velvet in terms of contiguity, but no information is available on accuracy of its assemblies [77]. Large memory requirements mean that assembly of non-hierarchical reads from large (e.g. mammalian) genomes is only practically feasible using a parallelization strategy such as that of ABYSS. However, a better solution might be to generate hierarchical sequence data from such genomes, as exemplified by refs. [104–107], though these methods are more laborious and may require larger quantities of expensive reagents.

Metrics of assembly quality

In order to compare the relative quality of one assembly against another, we need some metrics of quality. There are two dimensions of quality: contiguity and accuracy. Contiguity refers to the lengths of the contigs and/or the scaffolds. In practice, the set of contigs comprising an assembly will not be of uniform length, so the measure of contiguity is essentially a description of a distribution of lengths. Therefore, useful metrics might include the mean, median, minimum and maximum lengths. However, arguably the most useful summary statistic is the N_{50} length. N_{50} is calculated by first ordering all contigs (or scaffolds) by length and then summing their lengths (starting with the longest) until the sum exceeds 50% of the total length of all contigs. Alternatively, where the length of the target genome is known, then N_{50} calculation is sometimes based on 50% of the genome length rather than 50% of the sum of contig lengths [108]. The N_{50} contig (or scaffold) number is the number of contigs (or scaffolds) of at least N_{50} in length.

Clearly, a high degree of contiguity is desirable and may motivate the choice of method and parameter values. In practice, there is usually a trade-off between contiguity and accuracy; maximizing contiguity will likely mean a less accurate assembly [108]. For example, in a recent comparative study of chromosome 17 in two strains of mouse [74], rather than using any of the previously published programs for assembling their Illumina 36 nt reads, the authors developed a new assembler, called Fuzzypath [74]. The authors demonstrated that Fuzzypath yielded significantly longer contigs than either Velvet or Abyss. Unfortunately, they did not attempt to measure the accuracies of the assemblies.

Several studies have devised metrics designed to reflect the accuracy or 'correctness' of an assembly. MacCallum and colleagues [65] assessed assemblies of short (36 and 26 nucleotides) Illumina reads from several bacteria and a fungus for which high-quality complete genome sequences had previously been determined by capillary sequencing. They considered two aspects: 'base accuracy' and 'mis-assembly rate' for assemblies generated by Allpaths, Velvet and EULER-SR [71, 72]. Base accuracy referred to the frequency of calling the correct nucleotide at a given position in the assembly and equivalent to a PHRED quality score [109]. Mis-assembly rate refers to the frequency of rearrangements, significant insertions, deletions and inversions. MacCallum and colleagues [65] estimated both rates from alignments of 10 kb chunks of the Illumina assemblies against the previously published reference sequences. They calculated base accuracy from the rate of exact matches between Illumina-based assembly and reference sequence in the alignments and counted mis-assembly rate as the fraction of the assembly that fell within 10 Kb chunks that were <99% identical to the reference.

Apart from benchmarking and proof-of-principle studies, typically no suitable reference is available. Therefore, there is a need for assembly-accuracy metrics that are not based on alignment to a reference sequence. Of course, this issue is not restricted to assemblies built from short reads; it applies equally to assemblies of long reads and hybrid assemblies of multiple data types. One approach to validating hybrid assemblies is to check for consistency between paired reads from two independent data sets, e.g. Illumina versus capillary sequence datasets (e.g. [35]). Internal consistency can be checked in a pure short-read assembly, e.g. aligning original read-pairs against the assembly and checking the frequency of incongruence and regions of unusual coverage depth. Some such methods have been implemented in the 'amosvalidate' software [110]. A graphical visualization tool such as Tablet [111] can be invaluable for exploring assembly quality. Another common approach, used in the giant panda genome project [40], is to compare the assembly to existing independently obtained sequences (mRNA, cloned genes, etc.) from the same organism in the public databases. If sequences are not available from the same organism, then a set of highly conserved sequences from a set of related organisms can be used to interpolate sequences that should be conserved in the newly sequenced genome. For example, we have prepared a list of highly conserved Pseudomonas gene sequences against which we compare our *de novo* assemblies of genomes from this genus [43].

Assembling transcriptomic data

Most *de novo* DNA sequence assembly were designed primarily to handle genomic sequence. However, it is now clear that even a complete genome sequence is not enough; the behaviour and biological state of a cell largely depends on its transcriptome, i.e. the repertoire of protein-coding messenger RNAs and other transcripts. In practice, rather than directly sequencing RNA, the transcriptome is reverse transcribed into cDNA, which serves as the substrate for sequencing. The transcriptome differs from the genome in several important respects. Assembly of transcript sequences (for example, expressed sequence tags, ESTs) presents different challenges from the assembly of genomic sequence. Transcription is discontinuous, leading to much less contiguity in the transcriptome than the genome. Repeat sequences are less of a problem in the transcriptome; ESTs come from only a subset of the genome and are enriched for protein-coding sequences that tend to be less repetitive than intergenic regions than in the genome. The transcriptome has an additional complication in that a single genomic region can be transcribed into several different isoforms due to multiple transcription start and end sites and differential splicing. This situation is further complicated by contamination of the cDNA library by genomic DNA. Despite these challenges, Birol and colleagues [112] successfully assembled a transcriptome from a human cancer cell line using the ABySS assembler. Their cDNA library had been normalized using a DNA molecular denaturization and re-association method. Un-normalized cDNA would display extreme sampling bias, with a few highly expressed transcripts dominating the sample and rare transcripts being below the limits of detection. Even normalized cDNA shows some sampling bias, so Birol and colleagues chose a very low threshold depth-of-coverage value ($2\times$, compared with a default value of $5\times$) for trimming of (false) branches from the de Bruijn graph. They also took care to record (rather than simply discard) 'bubble' structures in the de Bruijn graph, which might represent alternative isomforms. As such they were able to discover examples of novel transcripts with skipped or retained exons, alternative $5'$ splicing and other novel isoforms [112]. Meanwhile, Zerbino and colleagues have begun developing a new assembler specifically optimized for transcript-derived short reads [76]. This tool, called Oases, is based on the Velvet assembler.

New assembly approaches to overcome limitations of short reads

In practice, the whole-genome shotgun assembly of short sequence reads has yielded poorer quality assemblies than assembly of longer capillary reads (and, indeed, 454 reads). Improvements in sequence assembly can be made by assembling two or more paired-read libraries with different insert lengths. However, several recent reports have demonstrated much-improved assemblies using a hierarchical approach to sequencing large genomes, combining novel *in vitro* protocols with *in silico* innovations. Young and colleagues [106] partitioned the fly genome into a series of eight overlapping libraries of fragments by digesting genomic DNA with two different restriction enzymes and fractionating the digested fragments into four size-classes. The sequence complexity within each library was significantly decreased compared with the whole genome. Each library was computationally assembled (using Velvet) and then the eight assemblies were merged (using Minimus). This combined assembly was significantly more contiguous than an assembly than the equivalent whole-genome shotgun assembly, having a N_{50} of 4.2 kb compared to just 1.4 kb for the whole-genome shotgun assembly. Their combined assembly was of adequate quality for comparative genomics studies with other fly genomes and the use of such libraries is a scalable strategy that should be applicable for cost-effective sequencing of mammalian genomes too.

A team from University of Washington took a rather different approach to exploiting in vitro library construction [105]. They proposed a strategy of 'sub-assembly' whereby they first sheared genomic DNA (from the bacterium Pseudomonas aeruginosa) into fragments of around 550 nucleotides. Fragments were then PCR-amplified along with a 20-nucleotide tag sequence. Effectively each unique 550-nucleotide fragment is labelled with a unique 20-nucleotide tag. Tagged fragments were then concatenated with each other, randomly sheared and PCR-amplified once more, this time incorporating a so-called 'breakpoint-adjacent adaptor' at the end of the fragment furthest from the 20-nucleotide label. This produced a library of fragments in which every short fragment is linked to a label that indicates which 550-nucleotide it came from. They performed paired-read sequencing on this library using the Illumina Genome Analyser II such that for each read-pair, one read contained the label tag and the other contained the genomic sequence adjacent to the breakpoint-adjacent adaptor. Thus, they could divide their short sequence reads into a series of pools, each originating from a single 550-nucleotide genomic fragment and assemble each independently. The result of these assemblies is essentially a set of 550-nucleotide sequence reads, comparable with the read-lengths obtained by 454 or capillary sequencing but generated on the Illumina platform for a much lower cost. In fact, these sub-assembled 'reads' will likely have a much lower error-rate than reads from the other platforms since they originate from the consensus of multiple overlapping reads. In principle, this kind of approach could be scaled up to larger fragment sizes and much larger genome sizes.

A third sample preparation technique was The Long March, proposed by Sorber and colleagues [107]. This approach is conceptually similar to the classical strategy of nested deletions [113]. During preparation for sequencing by Illumina 454 and SOLiD, target fragments are first ligated to special oligonucleotide sequencing adapters. Sorber and colleagues [107] modified Illumina sequencing adapters such that they contained a recognition site for a specific restriction enzyme (GsuI) that cleaves DNA at a site 14 nucleotides distal to the recognition site. Thus, on exposing their library to the restriction enzyme, the library fragments were cleaved resulting in partial deletion of the end of the read. Of course, the cleavage also resulted in loss of the adapter, so they had to re-ligate the cleaved fragments to fresh adapter oligonucleotides. They iteratively repeated this procedure of restriction digestion and re-ligation for two further rounds. They used slightly different adapter sequences for each of the three rounds (a technique known as *barcoding*) so that the adapter sequence could be used to identify from which round a given DNA sequence originated even after pooling together samples from all rounds. The result of these iterative rounds of digestion and ligation was a nested set of sub-libraries for Illumina sequencing. The same principle could be applied to other sequencing technologies and different restriction enzymes could be chosen to vary the length of the deletion. The advantage of The Long March is that it yields a greater degree of overlap between fragments (from different rounds of deletion) than would be obtained if, as the authors put it, 'chance is relied upon to produce reads with sufficient overlap for assembly' [107]. The authors did not explicitly test whether their data yielded improved assemblies by performing *de novo* assembly; however, based on assembly by alignment to a reference genome sequence, they convincingly demonstrated that The Long March gave broader and more even coverage. Given sufficient depth of sequence coverage and truly random sampling, degree of read-overlap should not be a limiting factor. Nested deletion approaches such as The Long March might be a good solution to overcoming the sequencing biases that sometimes limit assembly quality.

Strategies for filtering sequence reads prior to assembly

Although the commonly used assemblers are remarkably robust to sequence errors, it is advisable to filter short-read sequence data prior to assembly. The phrase 'garbage in garbage out' (GIGO) holds here, though some of the more conservative assemblers (e.g. Velvet) might be better described by 'garbage-in, nothing-out'; their consistency and error-elimination algorithms mean that they tend to generate no assembly rather than a very inaccurate assembly. In either case, input of large quantities of erroneous sequence reads will also lead to a significantly increased memory footprint; error-containing reads generate additional vertices and erroneous paths in the de Bruijn graph; a bigger graph requires more memory, which is often the limiting resource.

As a minimum, when assembling Illumina short sequence reads, we routinely discard all reads that contain any ambiguities ('Ns'). We also discard any homopolymer reads (e.g. strings of 'As') and reads that contain adapter sequences. The benefits of these filtering steps have not been systematically analysed, but anecdotally, we find that such filtering leads to better assemblies and less memory usage during assembly. The Illumina data analysis pipeline offers the option of 'purity filtering', which eliminates sequence reads that might originate from mixtures of more than one template sequence due to close proximity on the flowcell. Many sequencing centres routinely apply this filter prior to *de novo* assembly [55]. There is some controversy over the value of further filtering short sequence reads according to their quality scores. Prior to assembling long sequence reads, it is usual to trim the 5′ and 3′ ends of the reads to remove vector sequences and/or poor quality base calls. Base calling in long sequence reads is most commonly performed by the PHRED software [109, 113, 114], which assigns a quality score to each nucleotide in the read according to the estimated probability of an error. The Illumina data analysis pipeline provides similar quality scores, which are encoded as ASCII characters in the FASTQ sequence format that is commonly used for Illumina sequence output. In principle, then, it is possible to filter data by removing sequence reads having poor quality scores or to trim the reads to remove poor-scoring regions of reads. When adopting a trimming or filtering strategy it should be noted that there are in fact three different variants of the quality-score encoding [114]; it is important to know which variant the data uses. In our anecdotal experience, score-based trimming and filtering does not yield improvements in assembly quality when using Velvet; in fact it usually yields poorer assemblies (see Figure 3). However, it is possible that with some assembly algorithms and under some circumstances, filtering and trimming might have value. This issue deserves further systematic analysis.

Deposition of sequence into repositories

Having generated a sequence assembly that will form the basis for further analysis (comparison with other genomes, gene discovery, etc.) it will be essential, at some point, to submit the data to a public repository such as GenBank. Care needs to be taken and some post-processing of the assembly will probably be required. It may be helpful to illustrate this by briefly describing some issues that we encountered when submitting Velvet-derived assemblies of Illumina paired short reads to GenBank. Similar issues will arise from assemblies generated by other programs too. First, it is important to realize that the raw output from Velvet consists of scaffolds (or 'supercontigs'), although the default filename is 'contigs.fa'. In early versions of Velvet, there were often runs of between one and nine Ns indicating a gap between two adjacent contigs. However, GenBank does not accept inter-contig gaps of shorter than 10 nucleotides. Recent versions of Velvet comply with GenBank in this respect. However, it is still necessary to split the scaffold sequences into their component contigs and also generate an AFG file, which specifies the position of each contig within its cognate scaffold. Also, the default identifiers that Velvet ascribes to each scaffold do not comply with GenBank's regulations, so they need to be renamed. These post-processing steps can be easily automated using a scripting language such as Perl, but it is important to perform all downstream analyses on the processed sequences rather than raw Velvet output in order to avoid inconsistencies with the publicly available version in GenBank.

CONCLUDING REMARKS

It is now feasible to generate, from short sequence reads only, complete genome sequence assemblies of at least good draft quality, even for large mammalian genomes, though these assemblies contain many gaps and are by no means 'finished'. The great majority of genome sequences published in the last couple of years have been based on long reads (capillary or 454 sequences) or mixtures of long and short reads. However, recent innovations in bioinformatics and *in vitro* library preparation make assembly of short reads increasingly tractable. With the prevailing trend towards increasing read-lengths in technologies such as Illumina, it is quite possible that in 5 years from now, *de novo* assembly of short reads will be obsolete as sequencing becomes dominated by new long-read technologies such as Pacific Biosciences' SMRT platform [115], which will present new challenges for sequence assembly. However, the current cohort of sequencers are generating enormous quantities of short-read data, not to mention backlogs of unanalysed data, and will continue to do so for some time to come. The accuracy and performance of short-read assemblers looks set to continue to advance, with the most popular software packages being actively developed. There may be a lag between algorithmic innovation and implementation. For example, a recent study [118] unified approaches based on de Bruijn graphs, bidirected graphs and network flow to develop an efficient (exact polynomial time) assembly algorithm. As the fruits of such theoretical labour are adopted, we can expect further improvements.

Key Points

- Short sequence reads can be used for *de novo* assembly of genomes and transcriptomes, even though they were originally envisaged as tools for resequencing applications where a reference sequence is available.
- *De novo* sequencing projects have usually relied on the inclusion of at least some longer sequence reads.
- Recent achievements using exclusively short reads culminated in the complete sequencing of the giant panda genome.
- Recent innovations in bioinformatics and in vitro library preparation promise to bring short read sequence assembly up to a comparable level with longer-read technologies with respect to assembly quality.
- There is a need for systematic comparisons of the available assembly algorithms and metrics of assembly quality.
- To obtain the best results it is important to consider the use of insert length libraries with different lengths.

Acknowledgements

We are grateful to Anirvan Sengupta for making the SOPRA assembler available to us prior to publication. We thank Yasubumi Sakakibara for sharing unpublished data from the *Bacillus subtilis* natto genome project. We are indebted to Murray Grant and Nick Talbot for their guidance and strategic support in establishing high-throughput sequence analysis facilities at the University of Exeter. This manuscript benefitted substantially from the constructive suggestions of the anonymous reviewers, to whom we are also grateful.

References

1. Ansorge WJ. Next-generation DNA sequencing techniques. *N Biotechnol* 2009;**25**:195–203.
2. Fox S, Filichkin S, Mockler TC. Applications of ultra-high-throughput sequencing. *Methods Mol Biol* 2009;**553**: 79–108.
3. Hall N. Advanced sequencing technologies and their wider impact in microbiology. *J Exp Biol* 2007;**210**:1518–25.
4. Holt RA, Jones SJ. The new paradigm of flow cell sequencing. *Genome Res* 2008;**18**:839–46.
5. Imelfort M, Edwards D. De novo sequencing of plant genomes using second-generation technologies. *Brief Bioinform* 2009;**10**:609–18.
6. MacLean D, Jones JD, Studholme DJ. Application of 'next-generation' sequencing technologies to microbial genetics. *Nat Rev Microbiol* 2009;**7**:287–96.
7. Mardis ER. Next-generation DNA sequencing methods. *Annu Rev Genomics Hum Genet* 2008;**9**:387–402.
8. Metzker ML. Sequencing technologies – the next generation. *Nat Rev Genet* 2010;**11**:31–46.
9. Morozova O, Marra MA. Applications of next-generation sequencing technologies in functional genomics. *Genomics* 2008;**92**:255–64.
10. Shendure J, Mitra RD, Varma C, Church GM. Advanced sequencing technologies: methods and goals. *Nat Rev Genet* 2004;**5**:335–44.
11. Varshney RK, Nayak SN, May GD, Jackson SA. Next-generation sequencing technologies and their implications for crop genetics and breeding. *Trends Biotechnol* 2009;**27**: 522–30.
12. Bentley DR, Balasubramanian S, Swerdlow HP, *et al.* Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* 2008;**456**:53–9.
13. Bentley DR. Whole-genome re-sequencing. *Curr Opin Genet Dev* 2006;**16**:545–52.
14. Margulies M, Egholm M, Altman WE, *et al.* Genome sequencing in microfabricated high-density picolitre reactors. *Nature* 2005;**437**:376–80.
15. Flicek P, Birney E. Sense from sequence reads: methods for alignment and assembly. *Nat Methods* 2009;**6**:S6–12.
16. Horner DS, Pavesi G, Castrignanò T, *et al.* Bioinformatics approaches for genomics and post genomics applications of next-generation sequencing. *Brief Bioinform* 2010;**11**: 181–97.
17. Imelfort M. Sequence comparison tools. In: Edwards D, Hansen D, Stajich J, (eds). *Bioinformatics: Tools and Applications*. London: Springer, 2009;13–37.
18. Jackman SD, Birol I. Assembling genomes using short-read sequencing technology. *Genome Biol* 2010;**11**:202.
19. Pop M. Genome assembly reborn: recent computational challenges. *Brief Bioinform* 2009;**10**:354–66.
20. Pop M, Salzberg SL. Bioinformatics challenges of new sequencing technology. *Trends Genet* 2008;**24**:142–9.
21. Illumina Inc. http://www.illumina.com (25 May 2010, date last accessed).
22. The SOLiDTM System: Next-Generation Sequencing. http://solid.appliedbiosystems.com (25 May 2010, date last accessed).
23. Complete Genomics. http://www.completegenomics.com (25 May 2010, date last accessed).
24. Batzoglou S, Jaffe DB, Stanley K, *et al.* ARACHNE: a whole-genome shotgun assembler. *Genome Res* 2002;**12**: 177–89.

25. AMOS. http://amos.sourceforge.net (25 May 2010, date last accessed).
26. Havlak P, Chen R, Durbin KJ, *et al*. The Atlas genome assembly system. *Genome Res* 2004;**14**:721–32.
27. Myers EW, Sutton GG, Delcher AL, *et al*. A whole-genome assembly of Drosophila. *Science* 2000;**287**:2196–204.
28. Huang X, Madan A. CAP3: A DNA sequence assembly program. *Genome Res* 1999;**9**:868–77.
29. Pevzner PA, Tang H, Tesler G. De novo repeat classification and fragment assembly. *Genome Res* 2004;**14**: 1786–96.
30. Huang X, Wang J, Aluru S, *et al*. PCAP: a whole-genome assembly program. *Genome Res* 2003;**13**:2164–70.
31. de la Bastide M, McCombie WR. Assembling genomic DNA sequences with PHRAP. *Curr Protoc Bioinformatics* 2007; Chapter 11:Unit11.4.
32. Wang J, Wong GK, Ni P, *et al*. RePS: a sequence assembler that masks exact repeats identified from the shotgun data. *Genome Res* 2002;**12**:824–31.
33. Mullikin JC, Ning Z. The phusion assembler. *Genome Res* 2003;**13**:81–90.
34. Swaminathan K, Alabady M, Varala K, *et al*. Genomic and small RNA sequencing of Miscanthus x giganteus shows the utility of sorghum as a reference genome sequence for Andropogoneae grasses. *Genome Biol* 2010;**11**:R12.
35. Diguistini S, Liao NY, Platt D, *et al*. De novo genome sequence assembly of a filamentous fungus using Sanger, 454 and Illumina sequence data. *Genome Biol* 2009;**10**:R94.
36. Huang S, Li R, Zhang Z, *et al*. The genome of the cucumber, Cucumis sativus L. *Nat Genet* 2009;**41**:1275–81.
37. Reinhardt JA, Baltrus DA, Nishimura MT, *et al*. De novo assembly using low-coverage short read sequence data from the rice pathogen Pseudomonas syringae pv. oryzae. *Genome Res* 2009;**19**:294–305.
38. Smits TH, Jaenicke S, Rezzonico F, *et al*. Complete genome sequence of the fire blight pathogen Erwinia pyrifoliae DSM 12163T and comparative genomic insights into plant pathogenicity. *BMC Genomics* 2010;**11**:2.
39. Kerstens HH, Crooijmans RP, Veenendaal A, *et al*. Large scale single nucleotide polymorphism discovery in unsequenced genomes using second generation high throughput sequencing technology: applied to turkey. *BMC Genomics* 2009;**10**:479.
40. Li R, Fan W, Tian G, *et al*. The sequence and de novo assembly of the giant panda genome. *Nature* 2010;**463**: 311–7.
41. Nishito Y, Osana Y, Hachiya T, *et al*. Whole genome assembly of a natto production strain *Bacillus subtilis* natto from very short read data. *BMC Genomics* 2010;**11**:243.
42. Studholme DJ, Ibanez SG, MacLean D, *et al*. A draft genome sequence and functional screen reveals the repertoire of type III secreted proteins of Pseudomonas syringae pathovar tabaci 11528. *BMC Genomics* 2009;**10**:395.
43. Clarke CR, Cai R, Studholme DJ, *et al*. Pseudomonas syringae strains naturally lacking the classical P. syringae hrp/hrc Locus are common leaf colonizers equipped with an atypical type III secretion system. *Mol Plant Microbe Interact* 2010;**23**:198–210.
44. Almeida NF, Yan S, Lindeberg M, *et al*. A draft genome sequence of Pseudomonas syringae pv. tomato T1 reveals a type III effector repertoire significantly divergent from that of Pseudomonas syringae pv. tomato DC3000. *Mol Plant Microbe Interact* 2009;**22**:52–62.
45. Green S, Studholme DJ, Laue BE, *et al*. Comparative genome analysis provides insights into the evolution and adaptation of *Pseudomonas syringae* pv. *aesculi* on *Aesculus hippocastanum*. *PloS One* 2010;**5**:e10224.
46. Rees DJG, Husselmann LHH, Celton J-M. De novo genome sequencing of the apple scab (Venturia inaequalis) genome, using Illumina sequencing technology. *PAG-XVII Plant and Animal Genomes XVII Conference*. Available online at: http://www.intl-pag.org/17/abstracts/P01_PAGXVII_013.html.
47. Parks M, Cronn R, Liston A. Increasing phylogenetic resolution at low taxonomic levels using massively parallel sequencing of chloroplast genomes. *BMC Biol* 2009;**7**:84.
48. Whitall JB, Syring J, Parks M, *et al*. Finding a (pine) needle in a haystack: chloroplast genome sequence divergence in rare and widespread pines. *Mol Ecol* 2010;**19**:100–14.
49. Worley KC, Gibbs RA. Genetics: decoding a national treasure. *Nature* 2010;**463**:303–4.
50. Whiteford N, Haslam N, Weber G, *et al*. An analysis of the feasibility of short read sequencing. *Nucleic Acids Res* 2005; **33**:e171.
51. Chaisson M, Pevzner P, Tang H. Fragment assembly with short reads. *Bioinformatics* 2004;**20**:2067–74.
52. Kingsford C, Schatz MC, Pop M. Assembly complexity of prokaryotic genomes using short reads. *BMC Bioinformatics* 2010;**11**:21.
53. Farrer RA, Kemen E, Jones JD, Studholme DJ. De novo assembly of the Pseudomonas syringae pv. syringae B728a genome using Illumina/Solexa short sequence reads. *FEMS Microbiol Lett* 2009;**291**:103–11.
54. Kozarewa I, Ning Z, Quail MA, *et al*. Amplification-free Illumina sequencing-library preparation facilitates improved mapping and assembly of (G+C)-biased genomes. *Nat Methods* 2009;**6**:291–5.
55. Quail MA, Kozarewa I, Smith F, *et al*. A large genome center's improvements to the Illumina sequencing system. *Nat Methods* 2008;**5**:1005–10.
56. Aury JM, Cruaud C, Barbe V, *et al*. High quality draft sequences for prokaryotic genomes using a mix of new sequencing technologies. *BMC Genomics* 2008;**9**:603.
57. Srivatsan A, Han Y, Peng J, *et al*. High-precision, whole-genome sequencing of laboratory strains facilitates genetic studies. *PLoS Genet* 2008;**4**:e1000139.
58. Pevzner PA, Tang H, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA* 2001;**98**:9748–53.
59. Pevzner PA, Tang H. Fragment assembly with double-barreled data. *Bioinformatics* 2001;**17**(Suppl 1):S225–33.
60. Butler J, MacCallum I, Kleber M, *et al*. ALLPATHS: de novo assembly of whole-genome shotgun microreads. *Genome Res* 2008;**18**:810–20.
61. Zerbino DR, Birney E. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res* 2008;**18**:821–9.
62. Pevzner PA, Tang H, Tesler G. *De novo* repeat classification and fragment assembly. *Genome Res* 2004;**14**:1786–96.
63. Simpson JT, Wong K, Jackman SD, *et al*. Birol I. ABySS: a parallel assembler for short read sequence data. *Genome Res* 2009;**19**:1117–23.

64. ABySS. http://www.bcgsc.ca/platform/bioinfo/software/abyss (25 May 2010, date last accessed).
65. Maccallum I, Przybylski D, Gnerre S, *et al*. ALLPATHS 2: small genomes assembled accurately and with high continuity from short paired reads. *Genome Biol* 2009;**10**: R103.
66. ALLPATHS. http://www.broadinstitute.org/science/programs/genome-biology/computational-rd/computational-research-and-development (25 May 2010, date last accessed).
67. CLC bio. http://www.clcbio.com (25 May 2010, date last accessed).
68. Curtain. http://code.google.com/p/curtain (25 May 2010, date last accessed).
69. Hernandez D, François P, Farinelli L, *et al*. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome Res* 2008;**18**: 802–9.
70. Edena. http://www.genomic.ch/edena.php (25 May 2010, date last accessed).
71. Chaisson MJ, Brinza D, Pevzner PA. De novo fragment assembly with short mate-paired reads: does the read length matter? *Genome Res* 2009;**19**:336–46.
72. Chaisson MJ, Pevzner PA. Short read fragment assembly of bacterial genomes. *Genome Res* 2008;**18**:324–30.
73. EULSER-SR. http://euler-assembler.ucsd.edu/portal (25 May 2010, date last accessed).
74. Sudbery I, Stalker J, Simpson JT, *et al*. Deep short-read sequencing of chromosome 17 from the mouse strains A/J and CAST/Ei identifies significant germline variation and candidate genes that regulate liver triglyceride levels. *Genome Biol* 2009;**10**:R112.
75. FuzzyPath. ftp://ftp.sanger.ac.uk/pub/zn1/fuzzypath (25 May 2010, date last accessed).
76. Oases. http://www.ebi.ac.uk/~zerbino/oases (25 May 2010, date last accessed).
77. Bryant DW Jr, Wong WK, Mockler TC. QSRA: a quality-value guided de novo short read assembler. *BMC Bioinformatics* 2009;**10**:69.
78. QSRA. http://qsra.cgrb.oregonstate.edu (25 May 2010, date last accessed).
79. Rausch T, Koren S, Denisov G, *et al*. A consistency-based consensus algorithm for de novo and reference-guided sequence assembly of short reads. *Bioinformatics* 2009;**25**: 1118–24.
80. SeqCons. http://www.seqan.de/projects/seqcons.html (25 May 2010, date last accessed).
81. Dohm JC, Lottaz C, Borodina T, Himmelbauer H. SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. *Genome Res* 2007;**17**:1697–706.
82. SHARCGS. http://sharcgs.molgen.mpg.de (25 May 2010, date last accessed).
83. Hossain MS, Azimi N, Skiena S. Crystallizing short-read assemblies around seeds. *BMC Bioinformatics* 2009;**10**:S16.
84. SHORTY. http://www.cs.sunysb.edu/~skiena/shorty/ (25 May 2010, date last accessed).
85. Li R, Zhu H, Ruan J, *et al*. De novo assembly of human genomes with massively parallel short read sequencing. *Genome Res* 2010;**20**:265–72.
86. Short Oligonucleotide Analysis Package. http://soap.genomics.org.cn.
87. Dayarian A, Michael TP, Sengupta AM. SOPRA: an algorithm for high quality de novo assembly of paired reads via statistical optimization. In press.
88. SOPRA. http://www.physics.rutgers.edu/~anirvans/SOPRA/ (25 May 2010, date last accessed).
89. Warren RL, Sutton GG, Jones SJ, Holt RA. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics* 2007;**23**:500–1.
90. SSAKE. http://www.bcgsc.ca/platform/bioinfo/software/ssake (25 May 2010, date last accessed).
91. Schmidt B, Sinha R, Beresford-Smith B, Puglisi SJ. A fast hybrid short read fragment assembly algorithm. *Bioinformatics* 2009;**25**:2279–80.
92. Taipan. http://taipan.sourceforge.net (25 May 2010, date last accessed).
93. Jeck WR, Reinhardt JA, Baltrus DA, *et al*. Extending assembly of short DNA sequences to handle error. *Bioinformatics* 2007;**23**:2942–4.
94. VCAKE. http://sourceforge.net/projects/vcake (25 May 2010, date last accessed).
95. Zerbino DR, McEwen GK, Margulies EH, Birney E. Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. *PLoS One* 2009;**4**:e8407.
96. Velvet. http://www.ebi.ac.uk/~zerbino/velvet/ (25 May 2010, date last accessed).
97. Myers EW. Toward simplifying and accurately formulating fragment assembly. *J Comput Biol* 1995;**2**:275–90.
98. Idury RM, Waterman MS. A new algorithm for DNA sequence assembly. *J Comput Biol* 1995;**2**:291–306.
99. Pevzner PA. 1-Tuple DNA sequencing: computer analysis. *J Biomol Struct Dyn* 1989;**7**:63–73.
100. Sommer DD, Delcher AL, Salzberg SL, Pop M. Minimus: a fast, lightweight genome assembler. *BMC Bioinformatics* 2007;**8**:64.
101. Nagarajan N, Read TD, Pop M. Scaffolding and validation of bacterial genome assemblies using optical restriction maps. *Bioinformatics* 2008;**24**:1229–35.
102. Velvet-users – news for Velvet users. http://listserver.ebi.ac.uk/mailman/listinfo/velvet-users (25 May 2010, date last accessed).
103. Sundquist A, Ronaghi M, Tang H, *et al*. Whole-genome sequencing and assembly with high-throughput, short-read technologies. *PLoS One* 2007;**2**:e484.
104. Hiatt JB, Patwardhan RP, Turner EH, *et al*. Parallel, tag-directed assembly of locally derived short sequence reads. *Nat Methods* 2010;**7**:119–22.
105. Young AL, Abaan HO, Zerbino D, *et al*. A new strategy for genome assembly using short sequence reads and reduced representation libraries. *Genome Res* 2010;**20**:249–56.
106. Sorber K, Chiu C, Webster D, *et al*. The long march: a sample preparation technique that enhances contig length and coverage by high-throughput short-read sequencing. *PLoS One* 2008;**3**:e3495.
107. Salzberg SL, Yorke JA. Beware of mis-assembled genomes. *Bioinformatics* 2005;**21**:4320–1.
108. Ewing B, Green P. Base-calling of automated sequencer traces using phred. II. Error probabilities. *Genome Res* 1998;**8**:186–94.

109. Phillippy AM, Schatz MC, Pop M. Genome assembly forensics: finding the elusive mis-assembly. *Genome Biol* 2008;**9**:R55.

110. Milne I, Bayer M, Cardle L, *et al*. Tablet–next generation sequence assembly visualization. *Bioinformatics* 2010; **26**:401–2.

111. Birol I, Jackman SD, Nielsen CB, *et al*. De novo transcriptome assembly with ABySS. *Bioinformatics* 2009;**25**: 2872–7.

112. Hunkapiller T, Kaiser RJ, Koop BF, Hood L. Large-scale and automated DNA sequence determination. *Science* 1991; **254**:59–67.

113. Ewing B, Hillier L, Wendl MC, Green P. Base-calling of automated sequencer traces using phred. I. Accuracy assessment. *Genome Res* 1998;**8**:175–85.

114. Cock PJ, Fields CJ, Goto N, *et al*. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res* 2010; **38**:1767–71.

115. Eid J, Fehr A, Gray J, *et al*. Real-time DNA sequencing from single polymerase molecules. *Science* 2009;**323**:133–8.

116. Feil H, Feil WS, Chain P, *et al*. Comparison of the complete genome sequences of Pseudomonas syringae pv. syringae B728a and pv. tomato DC3000. *Proc Natl Acad Sci USA* 2005;**102**:11064–9.

117. Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res* 2008;**18**:1851–8.

118. Medvedev P, Brudno M. Maximum likelihood genome assembly. *J Comput Biol* 2009;**16**:1101–16.