COMPUTATIONAL TOOLS

Sequence assembly demystified

Niranjan Nagarajan[1] and Mihai Pop[2]

Abstract | Advances in sequencing technologies and increased access to sequencing services have led to renewed interest in sequence and genome assembly. Concurrently, new applications for sequencing have emerged, including gene expression analysis, discovery of genomic variants and metagenomics, and each of these has different needs and challenges in terms of assembly. We survey the theoretical foundations that underlie modern assembly and highlight the options and practical trade-offs that need to be considered, focusing on how individual features address the needs of specific applications. We also review key software and the interplay between experimental design and efficacy of assembly.

Previously the province of large genome centres, DNA sequencing is now a key component of research carried out in many laboratories. A number of new applications of sequencing technologies have been spurred by rapidly decreasing costs, including the study of microbial communities (metagenomics), the discovery of structural variants in genomes and the analysis of gene structure and expression. The length of the sequences generated by modern sequencing instruments is considerably shorter (hundreds to thousands of base pairs) than that of the genomes or genomic features being studied (which commonly span tens of thousands to billions of base pairs). Thus, many analyses start with the computational process of sequence assembly that joins together the many sequence fragments generated by the instrument. Biologists who need to assemble the sequencing data generated in their experiments are faced with the challenge of choosing, from a myriad of options, the assembly strategy and software best suited for their experiment. This choice is made harder by the rapid development of new assembly tools, which is driven by advances in sequencing technologies and the broader scope of applications for these technologies. Among recent innovations in sequence assembly are the use of memory-efficient data structures[1,2], the development of new *de novo* assembly strategies for data derived from metagenomic[3–5], single-cell[6,7] or transcriptome experiments[8,9] and the effective use of complementary information derived from multiple sequencing technologies[10,11] and/or paired-end data or mate-pair data[12,13].

All assembly approaches rely on the simple assumption that highly similar DNA fragments originate from the same position within a genome. The similarity between DNA sequences is then used to 'stitch' together the individual fragments into larger contiguous sequences (contigs), thereby recovering the information lost during the sequencing process. The assembly process is complicated by the fact that, in many cases, this underlying assumption is incorrect. For example, genomic repeats — segments of DNA repeated in an almost identical form throughout a genome — yield fragments with highly similar sequences that originate from different places in the genome. Similarly, in transcriptome or metagenomic samples, nearly identical sequences may originate from different transcripts or genomes within the sample. How much an assembler is confused by such artefacts primarily depends on the length of the sequences that are read by the sequencing instrument, as repetitive regions shorter than a sequencing read can be automatically resolved[14]. The accuracy of the sequence data also has an important role: the more errors an assembler is willing to tolerate within the data, the more similar distinct regions of a genome will appear to the assembler. Broadly speaking, segments of the genome that diverge by less than the error rate of the sequencing instrument cannot easily be distinguished by an assembler.

In this Review, we begin by providing an overview of the principles underlying modern assembly tools as well as the engineering trade-offs involved in designing them. We discuss the value of experimental design for successful assembly as well as the ongoing work in the important field of assembly evaluation. We then discuss trade-offs in the context of the most common applications of sequence assembly. Our goal is to provide non-specialist readers with sufficient background to make informed choices in the use of assembly techniques.

Paired-end data
Data from a pair of reads sequenced from ends of the same DNA fragment. The genomic distance between the reads is approximately known and is used to constrain assembly solutions. See also 'mate-pair read'.

Mate-pair data
Data from a pair of reads sequenced from the same circularized DNA fragment. The circularization step allows for larger fragments sizes to be used. They provide the same information as paired-end reads to the assembler.

Contiguous sequence
(Contig). A sequence reconstructed by assembling together multiple reads.

[1]*Computational and Systems Biology, Genome Institute of Singapore, 138672 Singapore.*
[2]*Department of Computer Science, University of Maryland, College Park, Maryland 20742, USA.*
Correspondence to M.P.
e-mail: mpop@umiacs.umd.edu

doi:10.1038/nrg3367
Published online 29 January 2013; corrected online 5 February 2013

Modern sequence assemblers

Mathematical analyses of sequence assembly, which dates back to the pioneering work of Esko Ukkonen in the 1980s[15,16], revealed the fundamental difficulty of reconstructing a genome from sequenced fragments. Depending on the relationship between the length of reads and the length of repeats in the DNA being assembled (BOX 1), genome assembly can range from trivial (when all repeats are shorter than the read length) to computationally intractable (that is, finding the correct answer requires trying an exponential number of arrangements of reads, a task that cannot be achieved even on the best supercomputers) to impossible[14] (that is, the information contained within the reads is insufficient to identify the correct sequence reconstruction from an exponential number of equally good alternatives). Given the characteristics of currently available sequencing technologies (BOX 1), few projects fall into the category of 'trivial' (primarily small viral genomes or transcriptomes). In most cases, full genome sequences cannot be efficiently and reliably reconstructed from the data produced by the sequencing experiment; rather, assemblers produce a fragmented and often error-prone picture of the sequence being assembled.

Assembly paradigms. Assemblers are based on one of several different paradigms, such as greedy, overlap–layout–consensus (OLC), de Bruijn graph and string graph (introduced in BOX 2). The choice of approach depends on the characteristics of the data being assembled. For example, de Bruijn-graph-based approaches have been successful in assembling highly accurate short reads (<~100 bp, such as those generated by the Illumina Solexa technology; BOX 1), whereas overlap-based approaches (such as OLC or string graph) are mostly used for longer, more inaccurate data (>200 bp, such as Roche 454 and Sanger sequencing data; BOX 1). However, this is not an exclusive arrangement: de Bruijn graph assemblers have successfully been used with longer reads by using a pre-processing stage to correct sequencing errors[17,18], and efficient overlap-based assemblers for short reads have also been developed[19]. The choice of assembly paradigm on its own plays a marginal part in defining the performance and efficiency of an assembler.

As sequencing technologies evolve, assembly tools are adapting to cope with the features and scale of the data (TABLE 1). Whereas early sequence assemblers had to make do with sparse coverage, modern assemblers have to deal with the problems of plenty. The success of modern sequence assemblers has thus primarily been determined by their ability to face the twin challenges of engineering (that is, dealing with the scale of data) and analysis (that is, adapting to and exploiting specific features of the data).

Engineering challenges. Modern assemblers must be able to analyse large data sets efficiently, to handle sequencing errors and to capture the repeat structure of the genome correctly. Meeting these challenges has been key in defining assembly tools that have had an important impact on the field. For example, one of the first widely used short-read assemblers, Velvet[20], made a mark by showing that high-quality assemblies could be obtained from ultra-short reads (~30 bp) and high-coverage data sets. This approach was extended to the assembly of large genomes in the program ABySS[21] and for the first *de novo* assembly of a mammalian genome entirely using short reads with the program SOAPdenovo[22]. SOAPdenovo is a memory-efficient assembler that also includes robust error correction (to reduce sequencing errors) and scaffolding modules (to leverage mate-pair data; see 'Analysis challenges' below).

The importance of error correction for de Bruijn-graph-based assembly, in particular, has led to the recent development of tools for this task[18,23,24] that serve as useful pre-processors in genome assembly applications. Whereas early short-read assemblers focused on

Read
The sequence generated by a sequencing machine from a DNA fragment.

Overlap
The relationship between two reads, the ends of which have highly similar sequences. The minimum length allowed for the corresponding sequence is an important parameter in assembly.

Box 1 | **Sequencing and mapping technologies**

A full survey of sequencing technologies is beyond the scope of this article. The table below compares the currently available technologies in terms of several characteristics of importance for genome assembly: read length, error rate and the ability to generate paired-end reads natively. Note that potentially all sequencing technologies can be used to sequence mate-pair libraries obtained by the circularization of long DNA fragments[91]. Furthermore, long-range linking information can be obtained from genome-mapping technologies, such as optical mapping[92].

For most technologies, read lengths and error rates depend on the specific characteristics of the sequencing experiment. The values provided in the table are those that are encountered in typical recent projects.

Technology	Read length (bp)	Error rate	Native paired-end read support	Refs
ABI/Solid	75	Low (~2%)	Yes	93
Illumina/Solexa	100–150	Low (<2%)	Yes	94
IonTorrent	~200	Medium (~4%)*	No	94
Roche/454	400–600	Medium (~4%)*	No	94
Sanger	Up to ~2,000 bp	Low (~2%)	Yes	
Pacific Biosciences	Up to ~15,000‡	High (~18%)	Yes (in strobe read mode)	39

*454 and Ion Torrent technologies are prone to errors in homopolymer regions, which are segments of the genome in which the same nucleotide is repeated multiple times[94]. ‡Pacific Biosciences instruments produce reads with an exponential distribution of read lengths, only a few of which reach the multi-kb range[10,11].

de Bruijn graph approaches to avoid computing overlaps for large sets of reads (a computationally intensive task using the algorithms available at the time), the use of efficient search data structures (in particular, the FM index[25]) has dramatically improved the scalability of overlap-based approaches (for example, the assembler SGA[19]). The need to process increasingly large data sets (comprising tens to hundreds of millions of reads) efficiently has also led to the increased use of parallel processing (concurrent analysis on multiple computing nodes) for both de Bruijn-graph-based[21,26] and overlap-based[27] assemblers.

Furthermore, memory efficiency has also become an important area of focus for genome[1,19] and metagenome[28] assemblers and spurred the development of new approaches to construct and to process assembly graphs. Recently introduced techniques for reducing memory consumption have included the use of sparse graph representations[2], compressed graph data structures[1], Bloom filters[28] and the FM index for efficient overlap calculation[19]. This new class of memory-efficient assemblers allows the analysis of much larger data sets (such as soil microbiomes[28]) and allows a broader range of scientists to take advantage

Box 2 | **Assembly paradigms**

The strategies used by sequence assemblers can be organized into three major paradigms.

Greedy

The assembler always makes the choice with the greatest immediate benefit: for example, the assembler always joins the reads that overlap best, as long as they do not contradict the already constructed assembly. The choices made by the assembler are inherently local and do not take into account the global relationship between the reads. Most greedy assemblers include heuristics that are designed to avoid misassembling repetitive sequences. Many early assemblers, such as phrap and TIGR Assembler[95], relied on this paradigm, as do some more recent tools, such as VCAKE[96]. The greedy paradigm is, however, not widely used owing to the inherently local assembly process that cannot easily use global information (such as long-range mate-pair links) to resolve repetitive genomes.

Overlap–layout–consensus

The assembler starts by identifying all pairs of reads that overlap sufficiently well and then organizes this information into a graph containing a node for every read and an edge between any pair of reads that overlap each other. This graph structure allows the development of complex assembly algorithms that can take into account the global relationship between the reads. A variant of this approach — string graph — simplifies the global overlap graph by removing redundant information (transitive edges). This paradigm was made popular by the work of Gene Myers, embodied in Celera Assembler[44] and dominated the assembly world until the emergence of the new generation of short-read sequencing technologies. Concerns about the computational complexity of overlap computation have limited the application of the overlap–layout–consensus (OLC) approach until recently, when the assembler SGA[19] introduced a new approach based on efficient string indexing data structures.

De Bruijn graph

De Bruijn graph assemblers model the relationship between exact substrings of length *k* extracted from the input reads. Similarly to the OLC approach, the nodes in the graph represent *k*-mers, and the edges indicate that the adjacent *k*-mers overlap by exactly *k* – 1 letters (for example, the 5-mers ACTAG and CTAGT share exactly four letters). Whereas the reads themselves are not directly modelled in this paradigm, they are implicitly represented as paths through the de Bruijn graph. Most de Bruijn graph assemblers use the read information to refine the graph structure and to remove graph patterns that are not consistent with the reads. Also, as the de Bruijn graph approach is based on exact matches, error correction approaches (used both before and during assembly) are crucial for achieving high-quality assemblies. The de Bruijn approach was popularized by the assembler Euler[17] and has dominated the design of modern assemblers targeted at short-read sequencing data, such as Velvet[20], SOAPdenovo[22] and ALLPATHS[30]. De Bruijn graph assemblers are, however, stymied by sequencing errors and will probably decrease in importance as reads become longer and more inaccurate.

The figure highlights the interplay between read length, assembly paradigm and the repeat structure of the genome being assembled. Represented is a segment of the repeat graph of a genome comprising a repeat (R) and the flanking unique regions (A–D). Multiple traversals of this graph are possible, leading to different genome reconstructions (ARB, CRD; and ARD, CRB). The repeat graph itself is independent of the chosen assembly paradigm and models the inherent ambiguity introduced by genomic repeats. The goal of the assembler is to use the information contained in the reads to approximate and to resolve the structure of the repeat graph. For example, a long read (r1) spanning the entire repeat indicates that the repeat occurs in two genomic neighbourhoods: ARB and CRD. Similar information can be obtained from the mate pair r5–r6. Short reads (namely, r2, r3 and r4) do not provide sufficient information to disambiguate the repeat as both the ARB and ARD reconstructions are compatible with overlaps between r2 and r3, and r2 and r4, respectively. *k*-mers (information underlying de Bruijn graph assemblers) uniformly cover the reads (shown only for r1 for simplicity). Note that *k*-mer length correlates with the read overlaps that can be detected by the assembler: for example, if the *k*-mer size is longer than the overlap between reads r2 and r3 (shown by the grey box), the resulting de Bruijn graph will not be able to join the corresponding reads.

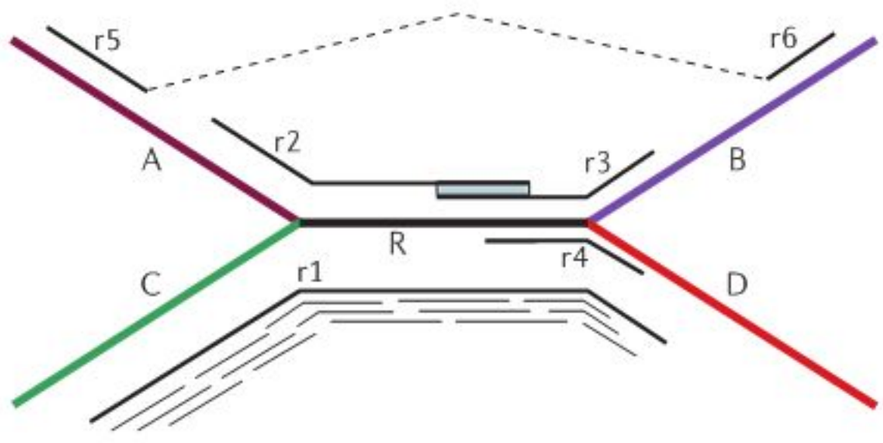


Table 1 | **Modern sequence assemblers: applications and sequencing technologies supported**

Assemblers	Technology	Availability	Notes	Refs
Genome assemblers				
ALLPATHS-LG	Illumina, Pacific Biosciences	ftp://ftp.broadinstitute.org/pub/crd/ALLPATHS/Release-LG	Requires a specific sequencing recipe (BOX 3)	40
SOAPdenovo	Illumina	http://soap.genomics.org.cn/soapdenovo.html	Also used for transcriptome and metagenome assembly	22
Velvet	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/velvet	May have substantial memory requirements for large genomes	20
ABySS	Illumina, SOLiD, 454, Sanger	http://www.bcgsc.ca/platform/bioinfo/software/abyss	Also used for transcriptome assembly	21
Metagenome assemblers				
Genovo	454	http://cs.stanford.edu/group/genovo	Uses a probabilistic model for assembly	66
MetaVelvet	Illumina, SOLiD, 454, Sanger	http://metavelvet.dna.bio.keio.ac.jp	Based on Velvet	4
Meta-IDBA	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/metaidba	Based on IDBA	5
Transcriptome assemblers				
Trinity	Illumina, 454	http://trinityrnaseq.sourceforge.net	Tailored to reconstruct full-length transcripts; may require substantial computational time	8
Oases	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/oases	Based on Velvet	72
Single-cell assemblers				
SPAdes	Illumina	http://bioinf.spbau.ru/en/spades		7
IDBA-UD	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/idba_ud	Based on IDBA	6

Note that only a few of the popular and freely available assemblers are included here for each application (a more complete list is provided in Supplementary information S1 (table)), and all of the listed assemblers (except Genovo) are based on de Bruijn graph construction. IBDA, Iterative De Bruijn graph short read Assembler.

of assembly tools without the need to acquire and to maintain high-performance computational infrastructures.

Analysis challenges. In addition to the engineering challenges outlined above, modern sequence assemblers have continued to develop new approaches for representing and analysing the assembly graph to assemble repeats correctly and to identify genomic variants distinguishing co-assembled haplotypes. One of the early pioneers in this direction was the program Euler[17], which popularized the de Bruijn paradigm for assembly as a way to model the repeat structure of a genome and introduced methods for using read and mate-pair information to resolve repeats[29]. Many of the techniques developed in Euler have been emulated and extended by most modern de Bruijn graph assemblers. Programs that followed, such as ALLPATHS[30] and Cortex[31], have developed new approaches for analysing the assembly graph structure to assemble repeats correctly and to identify genomic variants distinguishing co-assembled haplotypes[31].

Modern sequencing experiments typically generate mate-pair data: that is, information constraining the relative orientation and distance between pairs of reads. The ability to analyse this information to resolve repeats and to link together individual contigs into long-range scaffolds is an important area of improvement for assembly pipelines. Although many assembly tools include a scaffolding module, stand-alone software such as Bambus[32], SOPRA[33] and Opera[34] (which are often referred to as scaffolders) provides greater flexibility, particularly when combining data from different sequencing platforms. Tools for carrying out local assembly in low-coverage and repetitive regions (also called gap-filling or *in silico* finishing[35,36]) are valuable for validating and improving the assembly and are often a part of modern assembly pipelines.

Despite extensive mathematical analyses of the assembly problem[14–16], sequence assemblers continue to rely on heuristics and other ad hoc techniques rather than on rigorous algorithms with provable performance guarantees. This is in part owing to the difficulty of coming up with realistic mathematical models for assembly and in part owing to the sheer computational difficulty of the assembly problem. Most mathematical formulations of assembly suggest that finding the optimal assembly could require prohibitive computational resources[37]. Recent results indicate that optimal solutions may be possible for some assembly tasks, such as scaffolding[34] and finishing[36]; however, substantial work still remains to be done

Scaffolds
An ordered collection of contiguous sequences (contigs), the relative placement of which is typically inferred from mate-pair reads and other information. The sequence within the gaps between the contigs is usually not known.

before efficient and optimal 'black box' solutions for assembly are available for use by non-specialists. Until then, *in silico* assessment and experimental validation (see section below on 'Assessing assembly quality') are essential for verifying the data produced by assemblers before any biological conclusions can be drawn from them.

Experimental design

Users of sequencing technologies have many tunable parameters that they can control when designing a sequencing experiment. These parameters include the sequencing technology used, the length of the reads produced and the size and number of the mate-pair libraries. Each of the choices can affect the ability of an assembler to reconstruct the original DNA sequence correctly. As a pertinent example, Alkan *et al.*[38] have highlighted numerous errors in a recent *de novo* assembly of the human genome and argued for the continued development of new computational and experimental strategies that facilitate the complete and correct reconstruction of genomes. Below, we review several initial steps in this direction.

Read length has a fundamental impact on the complexity of assembly: the longer the reads, the fewer the repeats that confuse the assembly process. Read length is also one of the least 'tunable' parameters of the sequencing experiment. Although many sequencing technologies allow users to vary the length of the sequences generated (for example, by adjusting the number of cycles for which the instrument is run), the upper length limit is defined by fundamental limitations of the specific technology. Currently, the longest reads, up to about 14 kb[10,11], are produced by the Pacific Biosciences instruments[39]. This technology is increasingly used in the sequencing of bacterial genomes, in which most repeats (including ribosomal RNA operons) are usually shorter than ~6 kb. The long reads can be used by assemblers to resolve all genomic repeats and thereby to reconstruct entire genomes correctly[11]. Pacific Biosciences data, however, have high error rates, making it necessary to augment the experiment with data from more precise technologies[10,11]. Furthermore, the read lengths are exponentially distributed: whereas the longest reads can extend beyond 10 kb, most of the reads are much smaller and have a median size of only ~800 bp[10]. As only the long reads are useful as far as assembly is concerned, a substantial amount of data must be generated to ensure sufficient coverage of the genome by long reads. A recent study[11] reported that >200-fold coverage of the *Rhodobacter sphaeroides* genome was necessary to achieve just 1.6-fold coverage by reads longer than 2,000 bp. A different approach for increasing read length was proposed by the authors of ALLPATHS-LG[40], wherein short mate-pair libraries are constructed such that the paired reads overlap. For example, a 180 bp library of 100 bp Illumina reads would ensure that the mated reads overlap on average by 20 bp. The overlapping mates can then be stitched together into reads that are roughly twice as long as those produced by the sequencing instrument (BOX 3).

Mate-pair information is commonly used during assembly to resolve genomic repeats, to detect errors or structural variants and to scaffold together distant regions of the assembly. Wetzel *et al.*[12] showed that mate pairs are most able to resolve repeats (and thereby to increase the size and accuracy of the assembled contigs) if the sequencing experiment is 'tuned' to the repeat structure of the genome. Specifically, they propose a two-stage process in which an initial assembly is constructed from unmated reads to assess the size of the repeats; mate-pair libraries that best match the repeat sizes are then constructed. For example, the ideal mate-pair library for resolving a ribosomal RNA repeat of approximately 6 kb would span just a little over 6 kb, ensuring that the paired reads are anchored in the adjacent unique regions. Shorter mate pairs would not be able to disambiguate between multiple copies of the ribosomal operon nor would much longer mate pairs, such as commonly used fosmid libraries (35–40 kb), which may simultaneously span multiple operons.

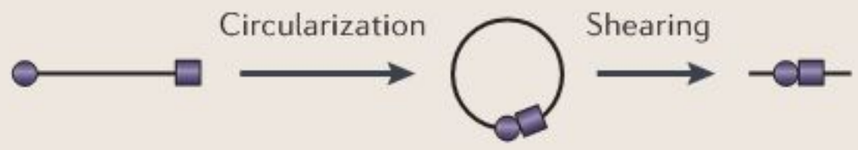
Bashir *et al.*[41] also explored the design of mate-pair sequencing experiments in the context of structural variation detection. They showed that two mate-pair libraries (a short one and a long one) are sufficient to optimize the ability to detect structural variants to within a level of resolution determined by the length of the short library. Their theoretical analysis ignored the presence of repeats, which affect the mapping of reads to the genome, but empirical results indicate that this simplification did not substantially affect the results. The authors considered the design of transcriptome sequencing projects as well[41]. Owing to the uneven distribution of transcript abundances in a sample, abundant transcripts can be reconstructed with limited sequencing depth, whereas the less abundant ones require substantially deeper coverage. They propose that it is possible to estimate, from an initial low-depth sequencing experiment, the relationship between depth of coverage and likelihood that a particular transcript is sampled by the sequencing data. This information can then be used to estimate the level of sequencing that is necessary to sample the entire transcriptome.

Finally, we would like to point out an increasing interaction between experimental design and the development of assembly approaches. This interaction is best exemplified by the ALLPATHS-LG assembler[40], which is specifically designed for the assembly of data generated according to a special Illumina-based 'recipe', including short overlapping fragment libraries and several long-range mate-pair libraries ranging in size from ~3 kb (short-jump library) to ~6 kb (long-jump library) to 40 kb (fosmid-jump library) (BOX 3). The success of the joint design of the assembler and associated sequencing experiment — ALLPATHS-LG arguably won the Assemblathon[42] and Gage[43] competitions — will hopefully spur a closer interaction between biologists and bioinformaticians in developing experimental strategies that generate data that can be most effectively used by the assembly software.

Library
A collection of paired-end or mate-pair reads derived from DNA fragments with a tightly controlled size range.

Depth of coverage
The average number of reads covering a particular base in the sequence being assembled.

Box 3 | **Sequencing recipes**



With the ALLPATHS-LG[40] assembler, the idea was introduced of tying the development of assembly algorithms and software with the development of a 'recipe' for the sequencing experiment. The benefits of the joint development of software and experiment are twofold: first, the assembler can more efficiently derive information from the data and thereby produce better assemblies; second, the developers no longer need to account for the diverse characteristics of data that might be generated in the sequencing experiment and can, therefore, focus their efforts on improving the accuracy and performance of the assembler. Below, we detail two sequencing recipes suggested by the developers of ALLPATHS-LG.

Mammalian genome recipe

This recipe is based on REF. 40.

45-fold coverage in 180 bp Illumina fragment library. The library size is chosen such that the paired reads overlap (that is, fragment size is smaller than twice the average read length). The assembler can merge the paired reads into a single long read that spans the entire DNA fragment, thereby effectively increasing the length of the reads. Longer reads are more effective in resolving repeats, leading to improved assembly. Note that the Illumina instrument can natively generate paired read data, albeit only from short DNA fragments.

45-fold coverage in 3 kb Illumina short-jump library. The term 'jump' refers to an experimental process that allows a DNA fragment to jump over a long segment of a genome. The process (as illustrated in the figure) often involves the circularization of DNA fragments of the desired size (3 kb in this case). The resulting circular segments are then sheared into small fragments that are suitable for sequencing. The mate-pair information is recovered by identifying, within the sequenced fragments, the junction between the ends of the original fragment (see the figure, in which the fragment ends were marked, for clarity, with a circle and square).

Fivefold coverage in 6 kb Illumina long-jump library (optional). Similar to the short-jump library except that the protocol is optimized for longer fragments.

Onefold coverage in 40 kb Illumina fosmid-jump library (optional). The fosmid library construction is similar to that for the short-jump libraries except that the amplification of the large DNA fragments requires transfection in an *Escherichia coli* vector.

Bacterial genome recipe

This recipe is based on REF. 11.

50-fold coverage in 160–220 bp Illumina fragment library. See above.

50-fold coverage in 1–3 kb Pacific Biosciences single reads. The Pacific Biosciences technology generates long (1–3 kb) to extremely long reads (up to ~15 kb), which are effective for resolving genomic repeats. Owing to the highly uneven distribution of read lengths (only a small fraction of the data represents long reads), high depth of coverage is necessary to ensure that sufficient numbers of the long reads are available to the assembler.

50-fold coverage in 2–10 kb Illumina jump library. See above.

Assessing assembly quality

Determining whether an assembly is correct and comparing the quality of different assemblies of the same data set are difficult given that the correct answer is usually not known (otherwise an assembly would be unnecessary). The output of assemblers is usually fragmented and often contains mistakes that range from small nucleotide changes to copy number changes in tandem repeats to large-scale rearrangements of the genome structure. All too often, scientists focus only on contiguity and ignore the correctness of the reconstructed sequences. Commonly used measures are total size, the number of contigs generated or the weighted median contig size (N50). In particular, the N50 size (which can be used appropriately to assess the contiguity of an assembly) is frequently misused in the literature by using the total assembly size as a proxy for genome size, leading to N50 numbers that cannot be compared across assemblies of the same genome (see REF. 42 for a discussion). Even if correctly used, however, N50 values rarely correlate with the actual quality of assembly, as demonstrated by recent assembly competitions[42,43]. N50 numbers are also meaningless in situations in which the goal is to reconstruct multiple sequences that are present in the sample at varying levels of abundance, such as in metagenomics or transcriptome assembly (see below).

To detect assembly errors, scientists have relied on independently derived information about the genome being assembled, such as mapping data[44–46], manually curated localized assemblies (for example, finished BAC sequences used to evaluate whole-genome assemblies[44,47]), transcriptome data[48] or the genomes of closely related organisms[49,50]. None of these approaches can completely verify the quality of an assembly: mapping data cannot detect single-nucleotide errors or short-range rearrangements; localized assemblies provide only partial information; and assembly errors cannot be easily distinguished from true biological differences between the assembly and assembled transcripts or related genomes (although careful evolutionary-based analyses can be used to distinguish

N50
A statistic used for assessing the contiguity of a genome assembly. The contigs in an assembly are sorted by size and added, starting with the largest. The size of the contig is reported that makes the total greater than or equal to 50% of the genome size.

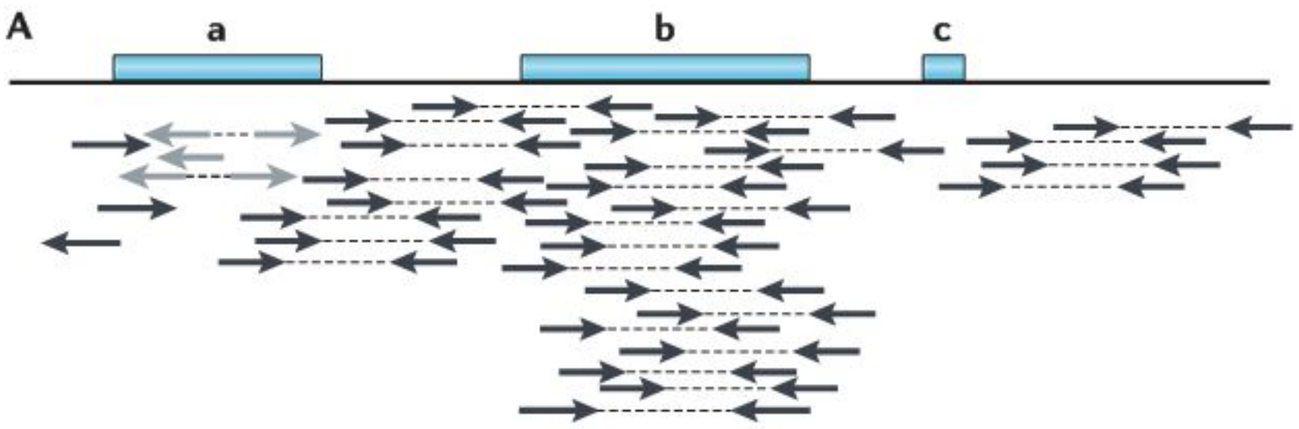


Figure 1 | **Methods for assembly validation. A** | Patterns in the alignment of reads along the assembled sequence, highlighting potential misassemblies: misoriented mate pairs, indicating a possible misjoin between unrelated genomic regions (**a**); a region with unusually deep coverage, indicating potential collapsed repeat (**b**); and a weak join, indicating a possible misjoin between unrelated genomic regions (**c**). **B** | Three assemblies (generated using ALLPATHS-LG, ABySS2 and SOAPdenovo) of the same data set (namely, *Staphylococcus aureus*) were compared using the feature response curve (FR-curve)[58] approach. The plot provides a visual representation of the trade-off between contiguity (cumulative assembly size on the *y* axis) and correctness (cumulative number of errors on the *x* axis). The assembled contiguous sequences (contigs) are considered in decreasing order of their sizes (the largest contigs occur at the bottom of the plot). The data were generated from the assemblies reported in REF. 43, and errors were estimated by alignment to the complete sequence of the genome. Note, however, that the analysis was primarily done for illustration purposes, and this figure should not be used to draw general conclusions about the relative performance of these assemblers. The curve corresponding to the ALLPATHS-LG assembly is always above and to the left of that for the SOAPdenovo assembly, indicating that the former is better (that is, it achieves higher contiguity for the same number of errors, or conversely, fewer errors for the same size). Also evident in the plot is the rapid accumulation of errors in small contigs (shown by the plateau on the right side of the curve). The curve corresponding to the ABySS2 assembly highlights an interesting artefact: this assembly contains more DNA than the other two but also greatly exceeds the actual size of the genome being assembled (as shown by the dashed line). Without prior knowledge of the genome size, this assembly may be preferred as it assembles more DNA with a similar number of errors as ALLPATHS-LG.

errors from true biological differences[49]). Intrinsic consistency measures have also been used to identify and to correct assembly errors[51]. Such measures include: the detection of regions with unusual depth of coverage that is either too high (possibly indicating the collapse of a repeat) or too low (possibly indicating an incorrect join between unrelated genomic regions); large numbers of mismatches between the assembled sequence and the sequencing reads (often found in collapsed repeats or misjoins); and inconsistent pairing of mated reads[52] (highlighting larger-scale genomic rearrangements).

Combinations of the approaches outlined above have been used to validate the quality of newly reconstructed genomes (for example, those of humans[38,47], *Drosophila melanogaster*[44], mice[53] and bonobos[54]), to compare multiple assembly tools[42,43] and to evaluate and to refine an assembly as it is being produced. Recent studies indicate that the nucleotide-level quality of assemblies widely varies depending on the complexity of the genome being assembled, ranging from about one error in every 100,000 bp for bacterial genomes[43] to roughly one error in every 1,000 bp for the human genome[43] and one error in 5,000 bp for the bonobo genome[54]. These numbers approach and even exceed the quality criteria (one error in 10,000 bp) established for the manually finished sequence generated by the Human Genome Project[55]. Although the sequence produced by assemblers is largely correct, more substantial errors occur within repeat regions[43,55], limiting the size of the genomic fragments that can be reliably reconstructed to a median of just a few thousand base pairs in complex genomes[43] (although long segments spanning millions of base pairs can be, and often are, reconstructed by modern assemblers). The iterative refinement of assemblies through mapping and recruitment of unassembled reads has been shown to be an effective approach for improving assembly quality. In particular, improvements come through localized assemblies within repeat-induced gaps, which overcome the inherent ambiguity caused by repeats and lead to improved contiguity[35].

Despite the importance of validation and the renewed interest in assembly technologies during the past few years, there are few computational tools that implement assembly validation techniques. These include: AMOSvalidate[51], a tool that carries out several of the consistency checks outlined above; GAV[56], a probabilistic approach for combining multiple accuracy measures; and the validation scripts used in the recent assembly competitions Assemblathon[42] and Gage[43]. Validation results can be visually explored with the assembly viewer Hawkeye[57], which plots the output of AMOSvalidate alongside the assembled pile-up of reads, allowing for manual evaluation of the result. An alternative representation is the feature response curve (FR-curve)[58], which provides an intuitive view of the contiguity versus errors trade-off across different assemblies of the same data set (FIG. 1).

Several recent studies have attempted to compare the performance of available genome assemblers, relying on high-quality gold standards (for example, complete or almost complete reference genomes)[43,59–61] or simulated data[42,59,60,62]. The results of these studies cannot easily be generalized to new genome projects as, for example, assembly tools behave differently depending

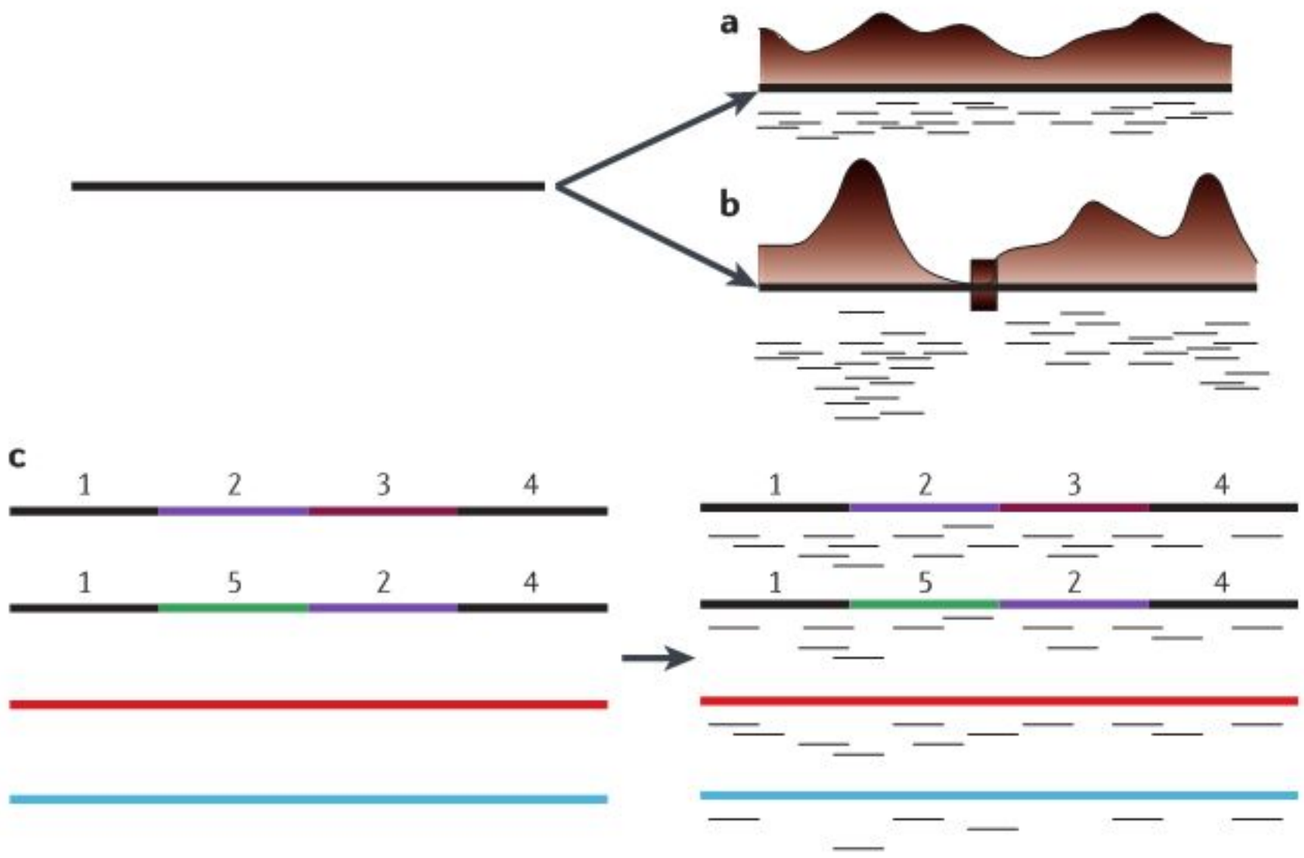


Figure 2 | **Comparison of isolate genome, single-cell and metagenomic assembly.** **a** | Depth-of-coverage histogram for an isolate genome sequencing project. Coverage is roughly uniform across the genome. **b** | Depth-of-coverage histogram for a single-cell project. Coverage is widely varying and genome regions may even be missed (red box). **c** | Metagenomic or transcriptomic project. Multiple genomes occur at different levels of coverage. Also note the similarities (denoted by colour and numbers) between recombined but closely related genomes. Reads originating from regions labelled with the same number appear to be identical to the assembler and cannot distinguish between the two genomes with genotypes 1–2–3–4 and 1–5–2–4, respectively.

on the specific structure of the sequencing experiment (in terms of the genome repeat structure, sequencing technology used, and so forth)[42,43,60,61]. Furthermore, these comparative studies highlight the absence of well-accepted validation measures: each study used a different collection of metrics and validation utilities, making it impossible to compare their respective results directly. There is a crucial need in the community for open-source validation tools that implement robust *de novo* methods for evaluating and comparing the correctness and contiguity of assemblies.

Evaluating the validation results produced by the approaches outlined above can be difficult as different assemblers choose a different trade-off between contiguity and correctness. As will be further detailed in the next section, this trade-off is affected by the specific application of the assembly software and even the ultimate goal of the sequencing process. For example, a study of the genes within an organism might sacrifice contiguity for accuracy, whereas studies of structural variation might tolerate small sequence errors to preserve long-range contiguity.

Applications of sequence assembly

Sequence assembly tools were originally developed for assembling whole genomes. The increased use of sequencing in new genomic applications has revealed the need for new assemblers that 'understand' the specific characteristics of the data being assembled. For example, in early assemblies of transcriptomic[63] and metagenomic data sets[64,65], existing genome assemblers were adapted with minor tweaks and changes. Recent work has emphasized the need for specialized assemblers that can effectively exploit the characteristics of the sequences that need to be reconstructed[8,66] (TABLE 1 and Supplementary information S1 (table)). Here we outline the most common applications of assembly tools and their specific characteristics that affect assembly strategies.

Whole-genome sequencing of isolate genomes. Isolate genome data are usually derived from few chromosomes present in the same number of copies within the cells being sequenced (notable exceptions are plasmids and organelles). As a result, assembly software can assume fairly even depth of coverage (FIG. 2a) and can use this information both to identify repeat regions (that is, regions in which the depth of coverage is unusually high) and to estimate the quality of the resulting assembly (see above). Furthermore, the high quality and quantity of DNA generated in isolate genome projects makes it easier to generate a broad range of mate-pair libraries; this information can be used to resolve repeats and increase assembly contiguity. Most assemblers available today (such as Velvet[20], SOAPdenovo[22], ABySS[21] and ALLPATHS[30]) were designed for the assembly of isolate genome data. Choosing from among these and many other similar tools primarily depends on the sequencing technology used (for example, Velvet can effectively assemble Illumina or Solid data, but Sanger sequences are likely to be better handled by a traditional assembler, such as the Celera Assembler[44]; TABLE 1) and the available computational resources (for example, ABySS was designed for distributed computing systems, such as computational grids, whereas SOAPdenovo is best suited on shared-memory systems, such as single servers with multiple processors).

Single-cell genomics. Obtaining sequence data from small numbers of cells usually requires aggressive whole-genome amplification techniques[67,68] that lead to chimeric sequences and highly non-uniform coverage of the genome being assembled[6,69] (FIG. 2b). As a result, statistical repeat detection and validation approaches developed for isolate genome assembly are not effective for single-cell genome sequencing data sets. Single-cell assemblers, such as IDBA-UD[6] and SPADes[7], thus have to rely on alternative approaches for detecting repeats and for correcting sequencing errors, leading to substantial improvements over generic genome assemblers[6,7].

Transcriptomics. Transcriptome data are typically derived from the total mRNA content of cells, comprising of a mixture of full-length and partial transcripts at various levels of abundance as well as the many possible splice forms of alternatively spliced genes. The sequences to be reconstructed thus have highly non-uniform coverage, even within the same transcript, and this problem is compounded by experimental limitations, such as the non-uniform amplification of mRNA[70]. Furthermore, cross-transcript repeats (for example, exons shared by multiple isoforms of the same gene) lead to ambiguity in

Isolate genome
The genome of a single organism isolated through culture, for which a substantial quantity of DNA can be obtained.

assembly. A successful strategy for coping with the uneven representation of transcripts involves running the assembly tool multiple times using a range of parameters that have been optimized for different levels of sequence abundance[9,71] (for example, longer *k*-mers allow accurate assembly of abundant transcripts, whereas shorter *k*-mers allow the assembly of low-coverage sequences). The resulting assemblies are then merged using ad hoc procedures to remove redundant contigs and to improve the assembly further.

Depending on the needs of their specific application, researchers may choose between assemblers that attempt to report just the most reliable transcripts, such as ABySS[63], and assemblers that attempt to maximize the number of transcripts reconstructed, such as Oases[72] and Trinity[8]. Furthermore, recent comparative studies[73,74] have revealed a trade-off between performance and accuracy. Constructing high-quality assemblies using, for example, Trinity[8] or Oases[72] requires substantial computational resources, whereas more efficient assemblers such as Trans-ABySS[9] may generate fragmented transcriptomes. These studies have focused only on the accuracy–efficiency trade-off and have not specifically evaluated other features of transcriptome assemblers, such as the ability to capture alternative isoforms or to reconstruct fusion genes, and these should serve as important points to investigate in future comparisons.

Metagenomics. Metagenomic data derive from the combined DNA content of viral, bacterial or eukaryotic communities (as opposed to individually isolated organisms). Many of the challenges encountered in the assembly of transcriptome data are also found in metagenomics, such as the varied depth of coverage across the individual chromosomes being assembled, the presence of cross-genome repeats (for example, ribosomal DNA or mobile elements shared by two or more organisms) and regions of genomic variation distinguishing otherwise identical genomes. Unlike transcriptome assembly, however, the depth of coverage within a chromosome is fairly even, and this information can be used to group together contigs that originate from the same genome[4,64]. To mitigate the effect of cross-genome repeats, scientists have relied on careful analysis of the assembly graph and the use of mate-pair information[3] and have also developed approaches for separating out individual genomes, which can then be assembled with traditional tools[4]. The latter approach can be affected by errors in the decomposition that lead to a higher rate of misassemblies[75].

The presence of multiple strains or similar species that differ owing to genomic rearrangements, recombination and mobile elements complicates the way in which assemblers format their output. Reconstructing individual genomes for strains of a species is typically impossible, as the genomic regions shared between strains (that is, those with nearly identical sequence) are often much longer than read lengths or commonly obtainable mate-pair sizes (FIG. 2c). Furthermore, there are no widely accepted approaches for constructing consensus genomic sequences along which genomic variants can be documented (although initial steps in this direction have been taken[3,5]). A further challenge is the sheer size of metagenomic data sets — sufficient sequencing must be carried out to ensure adequate representation of a reasonable fraction of genomes in a sample (for example, the Human Microbiome Project[76] estimated that approximately 15 Gb of sequencing are necessary to cover fully the genome of *Escherichia coli*, which is a minor member of gut communities[77]) — leading to the need for memory-efficient genome assemblers[28].

Although research on metagenomic assembly is still in its infancy, valuable scientific insights have already been derived[65,78]. These have come through the use of 'traditional' assemblers, such as SOAPdenovo[22], and newly developed tools that are specifically designed for metagenomic applications, including Bambus 2 (REF. 23), Meta-IDBA[5] and MetaVelvet[4].

Structural variations and haplotypes. An increasingly common application of sequence assembly is the study of structural variations and novel sequences[79] with respect to a reference genome. Most commonly, the reference genome is a closely related strain (such as a human individual being compared to the human genome reference). However, closely related organisms (for example, chimpanzees against the human reference[80]) can also be used, although no studies have yet evaluated the impact of the evolutionary divergence between the genomes on the effectiveness of variation discovery. Structural variation analyses start either with an assembly of the genome of interest, which is then compared to a reference sequence, or with identification of discordance in the direct alignment of unassembled mate-pair reads to the reference genome. The first approach may miss heterozygous events that are 'hidden' by the assembly process[81], whereas the second approach may fail in repeat regions or in the presence of complex rearrangement events. In the second approach, sequence assembly serves as a way to validate the structural variants predicted and to reconstruct the sequence surrounding the genomic break points[82–84]. A targeted, local assembly is done with relaxed criteria (that is, joining reads even if they overlap by only a small amount) to improve the ability to reconstruct sequences of uneven coverage; this approach is taken by BreakDancer[82].

It is important to note that sequence assembly tools (with a few notable exceptions[31,84]) typically reconstruct a linear consensus sequence and do not explicitly handle polyploidy. After a consensus sequence has been obtained and variant positions identified, a logically distinct set of tools — haplotype assemblers (for example, HapCompass[85] and HapCut[86]) — can try to string variations together to determine distinct haplotypes. A similar version of this problem is encountered in metagenomic or viral quasi-species data sets in which the number and abundance of haplotypes (that is, highly similar but distinct genomes) is variable and has to be estimated directly from the data[87,88]. Programs that are specifically tailored for this problem include ShoRAH[87], Vispa[89] and QuRe[90].

k-mers
Strings of *k* consecutive letters extracted from a longer sequence, such as a read or a reference assembly.

Conclusions

The rapid development of new sequencing technologies is being mirrored by the development of new genome assembly tools that are able to handle the characteristic features of the new technologies as well as the increased scope of genomic applications that rely on sequencing data. Despite the many new assembly tools becoming available on a monthly basis within the community, most of the advances in the field have been of a practical rather than a theoretical nature and have been targeted at engineering issues, such as memory consumption and the ability to handle new types of data. Despite continued and rapid advances in sequencing technologies, modern sequencing data carry limited information for use in assembly algorithms, and thus automated reconstruction of whole genomes is unlikely in the near future. An emerging trend in the field is the simultaneous development of assembly algorithms and sequencing experiments, allowing researchers to generate data that can most effectively inform the assembly process. Initial forays in this direction have occurred for both eukaryotic[40] and bacterial[11] assembly, for structural variation and transcriptome assembly[41] and have targeted both read-length[11,40] and mate-pair sizing[12,41]. We hope that this trend will lead to a tighter interaction between tool developers and the technology community. Future sequencing technologies should be evaluated not just through cost, throughput or length of reads, but also through their ability to inform bioinformatic analyses (including, but not limited to, assembly) of the resulting data. In other words, the technical costs of the sequencing experiment must be balanced against the effectiveness and ultimate cost of the downstream analysis process, which is often substantially higher in modern experiments.

New sequencing technologies (for example, Oxford Nanopore) have been announced that may generate substantially longer reads than is currently possible. These long reads may eliminate or substantially reduce the challenge posed by genomic repeats, raising the importance of other challenges faced by modern assemblers, in particular in the context of haplotype resolution and analysis of genomic variation in both eukaryotes and metagenomic data sets.

1. Conway, T. C. & Bromage, A. J. Succinct data structures for assembling large genomes. *Bioinformatics* **27**, 479–486 (2011).
2. Ye, C., Ma, Z. S., Cannon, C. H., Pop, M. & Yu, D. W. Exploiting sparseness in *de novo* genome assembly. *BMC Bioinformatics* **13** (Suppl. 6), S1 (2012).
3. Koren, S., Treangen, T. J. & Pop, M. Bambus 2: scaffolding metagenomes. *Bioinformatics* **27**, 2964–2971 (2011).
4. Namiki, T., Hachiya, T., Tanaka, H. & Sakakibara, Y. MetaVelvet: an extension of Velvet assembler to *de novo* metagenome assembly from short sequence reads. *Nucleic Acids Res.* **40**, e155 (2012).
5. Peng, Y., Leung, H. C., Yiu, S. M. & Chin, F. Y. Meta-IDBA: a *de novo* assembler for metagenomic data. *Bioinformatics* **27**, i94–i101 (2011).
6. Peng, Y., Leung, H. C., Yiu, S. M. & Chin, F. Y. IDBA-UD: a *de novo* assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics* **28**, 1420–1428 (2012).
7. Bankevich, A. *et al.* SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J. Comput. Biol.* **19**, 455–477 (2012).
This paper describes new assembly algorithms that are targeted at data generated in single-cell experiments through whole-genome amplification. The authors had to develop strategies for dealing with the highly uneven coverage of the data as well as numerous experimental errors.
8. Grabherr, M. G. *et al.* Full-length transcriptome assembly from RNA-seq data without a reference genome. *Nature Biotech.* **29**, 644–652 (2011).
Presented here is a collection of tools, called Trinity, for *de novo* assembly-based analysis of transcriptome data. This paper demonstrates that complete transcripts, including their splice forms, can be reconstructed from RNA-seq data.
9. Robertson, G. *et al. De novo* assembly and analysis of RNA-seq data. *Nature Methods* **7**, 909–912 (2010).
10. Koren, S. *et al.* Hybrid error correction and *de novo* assembly of single-molecule sequencing reads. *Nature Biotech.* **30**, 693–700 (2012).
11. Ribeiro, F. J. *et al.* Finished bacterial genomes from shotgun sequence data. *Genome Res.* **22**, 2270–2277 (2012).
12. Wetzel, J., Kingsford, C. & Pop, M. Assessing the benefits of using mate-pairs to resolve repeats in *de novo* short-read prokaryotic assemblies. *BMC Bioinformatics* **12**, 95 (2011).
13. Pham, S. K. *et al.* Pathset graphs: a novel approach for comprehensive utilization of paired reads in genome assembly. *J. Comput. Biol.* 17 Jul 2012 (doi:10.1089/cmb.2012.0098).
14. Nagarajan, N. & Pop, M. Parametric complexity of sequence assembly: theory and applications to next generation sequencing. *J. Comput. Biol.* **16**, 897–908 (2009).
An overview is provided here of the algorithmic challenges that underlie genome assembly; the paper has a specific focus on the interplay between read length and the size of repeats that can be correctly assembled.
15. Peltola, H., Soderlund, H. & Ukkonen, E. SEQAID: a DNA sequence assembling program based on a mathematical model. *Nucleic Acids Res.* **12**, 307–321 (1984).
16. Peltola, H., Sonderlund, H., Tarhio, J. & Ukkonen, E. in *IFIP 9th World Computer Congress* (ed. Mason, R. E. A.) 53–64 (North-Holland, 1983).
17. Pevzner, P. A., Tang, H. & Waterman, M. S. An Eulerian path approach to DNA fragment assembly. *Proc. Natl Acad. Sci. USA* **98**, 9748–9753 (2001).
18. Ronen, R., Boucher, C., Chitsaz, H. & Pevzner, P. SEQuel: improving the accuracy of genome assemblies. *Bioinformatics* **28**, i188–i196 (2012).
19. Simpson, J. T. & Durbin, R. Efficient *de novo* assembly of large genomes using compressed data structures. *Genome Res.* **22**, 549–556 (2012).
20. Zerbino, D. R. & Birney, E. Velvet: algorithms for *de novo* short read assembly using de Bruijn graphs. *Genome Res.* **18**, 821–829 (2008).
The Velvet assembler is the first widely used de Bruijn graph assembler, and this is the first paper to demonstrate that high-quality assembly of ultra-short reads is feasible.
21. Simpson, J. T. *et al.* ABySS: a parallel assembler for short read sequence data. *Genome Res.* **19**, 1117–1123 (2009).
The assembler described in this study, ABySS, is the first parallel genome assembler capable of assembling human-sized data sets.
22. Li, R. *et al. De novo* assembly of human genomes with massively parallel short read sequencing. *Genome Res.* **20**, 265–272 (2010).
23. Kelley, D. R., Schatz, M. C. & Salzberg, S. L. Quake: quality-aware detection and correction of sequencing errors. *Genome Biol.* **11**, R116 (2010).
24. Salmela, L. & Schroder, J. Correcting errors in short reads by multiple alignments. *Bioinformatics* **27**, 1455–1461 (2011).
25. Ferragina, P. & Manzini, G. in *Proc. 41st Annu. Symp. Foundations Comput. Sci.* 390–398 (2000).
26. Liu, Y., Schmidt, B. & Maskell, D. L. Parallelized short read assembly of large genomes using de Bruijn graphs. *BMC Bioinformatics* **12**, 354 (2011).
27. Xing, L. PASQUAL: parallel techniques for next generation genome sequence assembly. *IEEE Trans. Parallel Distrib. Syst.* 10 Aug 2012 (doi:10.1109/TPDS.2012.190).
28. Pell, J. *et al.* Scaling metagenome sequence assembly with probabilistic de Bruijn graphs. *Proc. Natl Acad. Sci. USA* **109**, 13272–13277 (2012).
29. Pevzner, P. A. & Tang, H. Fragment assembly with double-barreled data. *Bioinformatics* **17** (Suppl. 1), S225–S233 (2001).
This paper introduces the de Bruijn graph paradigm for assembly and the Euler assembler. The concepts described here have formed the basis for almost all de Bruijn-graph-based assemblers that are available in the community.
30. Butler, J. *et al.* ALLPATHS: *de novo* assembly of whole-genome shotgun microreads. *Genome Res.* **18**, 810–820 (2008).
31. Iqbal, Z., Caccamo, M., Turner, I., Flicek, P. & McVean, G. *De novo* assembly and genotyping of variants using colored de Bruijn graphs. *Nature Genet.* **44**, 226–232 (2012).
32. Pop, M., Kosack, D. S. & Salzberg, S. L. Hierarchical scaffolding with Bambus. *Genome Res.* **14**, 149–159 (2004).
33. Dayarian, A., Michael, T. P. & Sengupta, A. M. SOPRA: scaffolding algorithm for paired reads via statistical optimization. *BMC Bioinformatics* **11**, 345 (2010).
34. Gao, S., Sung, W. K. & Nagarajan, N. Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences. *J. Comput. Biol.* **18**, 1681–1691 (2011).
In this study, it is demonstrated that the genome scaffolding problem can be solved exactly for commonly encountered data despite the computational intractability of this problem. This paper also introduces the scaffolder Opera, which outperforms other stand-alone scaffolding packages.
35. Tsai, I. J., Otto, T. D. & Berriman, M. Improving draft assemblies by iterative mapping and assembly of short reads to eliminate gaps. *Genome Biol.* **11**, R41 (2010).
36. Gao, S., Bertrand, D. & Nagarajan, N. FinIS: improved *in silico* finishing using an exact quadratic programming formulation. *Lect. Notes Comput. Sci.* **7534**, 314–325 (2012).
37. Medvedev, P., Georgiou, K., Myers, G. & Brudno, M. Computability of models for sequence assembly. *Lect. Notes Comput. Sci.* **4645**, 289–301 (2007).

38. Alkan, C., Sajjadian, S. & Eichler, E. E. Limitations of next-generation genome sequence assembly. *Nature Methods* **8**, 61–65 (2011).
The many errors found in a *de novo* assembly of the human genome are highlighted here, and the authors argue for the continued development of experimental techniques aimed at fully reconstructing genomes.
39. Eid, J. *et al.* Real-time DNA sequencing from single polymerase molecules. *Science* **323**, 133–138 (2009).
40. Gnerre, S. *et al.* High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl Acad. Sci. USA* **108**, 1513–1518 (2011).
This paper introduces the ALLPATHS-LG assembler, which is the first assembler that is specifically designed in concert with a specific 'recipe' for the sequencing experiment.
41. Bashir, A., Bansal, V. & Bafna, V. Designing deep sequencing experiments: structural variation, haplotype assembly, and transcript abundance. *BMC Genomics* **11**, 385 (2010).
42. Earl, D. *et al.* Assemblathon 1: a competitive assessment of *de novo* short read assembly methods. *Genome Res.* **21**, 2224–2241 (2011).
The Assemblathon competition compared the performance of modern genome assemblers on a simulated human-sized diploid genome. The assemblies were contributed by the community, thus reflecting the best results that could be obtained with the corresponding assemblers. The paper also includes a detailed description of methods for validating the quality of the resulting assemblies.
43. Salzberg, S. L. *et al.* GAGE: a critical evaluation of genome assemblies and assembly algorithms. *Genome Res.* **22**, 557–567 (2012).
The GAGE competition compared the performance of several modern genome assemblers on real sequencing data from bacterial to eukaryotic genomes. The assemblies were carried out by the authors of the study, and the validation of the assemblies was done by comparison to known references for the genomes included. In addition, the paper provides full 'assembly recipes', which allow readers directly to reproduce the results presented.
44. Myers, E. W. *et al.* A whole-genome assembly of *Drosophila*. *Science* **287**, 2196–2204 (2000).
45. Zhou, S. *et al.* A whole-genome shotgun optical map of *Yersinia pestis* strain KIM. *Appl. Environ. Microbiol.* **68**, 6321–6331 (2002).
46. Nagarajan, N., Read, T. D. & Pop, M. Scaffolding and validation of bacterial genome assemblies using optical restriction maps. *Bioinformatics* **24**, 1229–1235 (2008).
47. Istrail, S. *et al.* Whole-genome shotgun assembly and comparison of human genome assemblies. *Proc. Natl Acad. Sci. USA* **101**, 1916–1921 (2004).
48. Zimin, A. V. *et al.* A whole-genome assembly of the domestic cow, *Bos taurus*. *Genome Biol.* **10**, R42 (2009).
49. Meader, S., Hillier, L. W., Locke, D., Ponting, C. P. & Lunter, G. Genome assembly quality: assessment and improvement using the neutral indel model. *Genome Res.* **20**, 675–684 (2010).
50. Gnerre, S., Lander, E. S., Lindblad-Toh, K. & Jaffe, D. B. Assisted assembly: how to improve a *de novo* genome assembly by using related species. *Genome Biol.* **10**, R88 (2009).
51. Phillippy, A. M., Schatz, M. C. & Pop, M. Genome assembly forensics: finding the elusive mis-assembly. *Genome Biol.* **9**, R55 (2008).
52. Huson, D. *et al.* in *Proc. First Int. Workshop Algorithms Bioinf.* 294–306 (2001).
53. Waterston, R. H. *et al.* Initial sequencing and comparative analysis of the mouse genome. *Nature* **420**, 520–562 (2002).
54. Prufer, K. *et al.* The bonobo genome compared with the chimpanzee and human genomes. *Nature* **486**, 527–531 (2012).
55. Blakesley, R. W. *et al.* An intermediate grade of finished genomic sequence suitable for comparative analyses. *Genome Res.* **14**, 2235–2244 (2004).
56. Choi, J. H. *et al.* A machine-learning approach to combined evidence validation of genome assemblies. *Bioinformatics* **24**, 744–750 (2008).
57. Schatz, M. C. *et al.* Hawkeye and AMOS: visualizing and assessing the quality of genome assemblies. *Brief. Bioinform.* 23 Dec 2012 (doi:10.1093/bib/bbr074).
58. Narzisi, G. & Mishra, B. Comparing *de novo* genome assembly: the long and short of it. *PLoS ONE* **6**, e19175 (2011).
59. Haiminen, N., Kuhn, D. N., Parida, L. & Rigoutsos, I. Evaluation of methods for *de novo* genome assembly from high-throughput sequencing reads reveals dependencies that affect the quality of the results. *PLoS ONE* **6**, e24182 (2011).
60. Lin, Y. *et al.* Comparative studies of *de novo* assembly tools for next-generation sequencing technologies. *Bioinformatics* **27**, 2031–2037 (2011).
61. Zhang, W. *et al.* A practical comparison of *de novo* genome assembly software tools for next-generation sequencing technologies. *PLoS ONE* **6**, e17915 (2011).
62. Barthelson, R., McFarlin, A. J., Rounsley, S. D. & Young, S. Plantagora: modeling whole genome sequencing and assembly of plant genomes. *PLoS ONE* **6**, e28436 (2011).
63. Birol, I. *et al. De novo* transcriptome assembly with ABySS. *Bioinformatics* **25**, 2872–2877 (2009).
64. Tyson, G. W. *et al.* Community structure and metabolism through reconstruction of microbial genomes from the environment. *Nature* **428**, 37–43 (2004).
65. Qin, J. *et al.* A human gut microbial gene catalogue established by metagenomic sequencing. *Nature* **464**, 59–65 (2010).
This is a large-scale catalogue of metagenomic data generated through *de novo* assembly of short read sequencing data. This paper is the first to demonstrate that metagenomic data can be effectively analysed through next-generation sequencing technologies.
66. Laserson, J., Jojic, V. & Koller, D. Genovo: *de novo* assembly for metagenomes. *J. Computat. Biol.* **18**, 429–443 (2011).
67. Dean, F. B. *et al.* Comprehensive human genome amplification using multiple displacement amplification. *Proc. Natl Acad. Sci. USA* **99**, 5261–5266 (2002).
68. Raghunathan, A. *et al.* Genomic DNA amplification from a single bacterium. *Appl. Environ. Microbiol.* **71**, 3342–3347 (2005).
69. Chitsaz, H. *et al.* Efficient *de novo* assembly of single-cell bacterial genomes from short-read data sets. *Nature Biotech.* **29**, 915–921 (2011).
70. Hansen, K. D., Brenner, S. E. & Dudoit, S. Biases in Illumina transcriptome sequencing caused by random hexamer priming. *Nucleic Acids Res.* **38**, e131 (2010).
71. Surget-Groba, Y. & Montoya-Burgos, J. I. Optimization of *de novo* transcriptome assembly from next-generation sequencing data. *Genome Res.* **20**, 1432–1440 (2010).
72. Schulz, M. H., Zerbino, D. R., Vingron, M. & Birney, E. Oases: robust *de novo* RNA-seq assembly across the dynamic range of expression levels. *Bioinformatics* **28**, 1086–1092 (2012).
73. Zhao, Q. Y. *et al.* Optimizing *de novo* transcriptome assembly from short-read RNA-seq data: a comparative study. *BMC Bioinformatics* **12** (Suppl. 14), S2 (2011).
74. Feldmeyer, B., Wheat, C. W., Krezdorn, N., Rotter, B. & Pfenninger, M. Short read Illumina data for the *de novo* assembly of a non-model snail species transcriptome (*Radix balthica*, Basommatophora, Pulmonata), and a comparison of assembler performance. *BMC Genomics* **12**, 317 (2011).
75. Charuvaka, A. & Rangwala, H. Evaluation of short read metagenomic assembly. *BMC Genomics* **12** (Suppl. 2), S8 (2011).
76. The Human Microbiome Project Consortium. A framework for human microbiome research. *Nature* **486**, 215–221 (2012).
77. Weinstock, G. M. Genomic approaches to studying the human microbiota. *Nature* **489**, 250–256 (2012).
78. The Human Microbiome Project Consortium. Structure, function and diversity of the healthy human microbiome. *Nature* **486**, 207–214 (2012).
79. Hajirasouliha, I. *et al.* Detection and characterization of novel sequence insertions using paired-end next-generation sequencing. *Bioinformatics* **26**, 1277–1283 (2010).
80. Newman, T. L. *et al.* A genome-wide survey of structural variation between human and chimpanzee. *Genome Res.* **15**, 1344–1356 (2005).
81. Khaja, R. *et al.* Genome assembly comparison identifies structural variants in the human genome. *Nature Genet.* **38**, 1413–1418 (2006).
82. Chen, K. *et al.* BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nature Methods* **6**, 677–681 (2009).
83. Chen, K. *et al.* BreakFusion: targeted assembly-based identification of gene fusions in whole transcriptome paired-end sequencing data. *Bioinformatics* **28**, 1923–1924 (2012).
84. Warren, R. L. & Holt, R. A. Targeted assembly of short sequence reads. *PLoS ONE* **6**, e19816 (2011).
85. Aguiar, D. & Istrail, S. HapCompass: a fast cycle basis algorithm for accurate haplotype assembly of sequence data. *J. Comput. Biol.* **19**, 577–590 (2012).
86. Bansal, V. & Bafna, V. HapCUT: an efficient and accurate algorithm for the haplotype assembly problem. *Bioinformatics* **24**, i153–i159 (2008).
87. Eriksson, N. *et al.* Viral population estimation using pyrosequencing. *PLoS Comput. Biol.* **4**, e1000074 (2008).
88. Prosperi, M. C. *et al.* Combinatorial analysis and algorithms for quasispecies reconstruction using next-generation sequencing. *BMC Bioinformatics* **12**, 5 (2011).
89. Astrovskaya, I. *et al.* Inferring viral quasispecies spectra from 454 pyrosequencing reads. *BMC Bioinformatics* **12**, (Suppl. 6), S1 (2011).
90. Prosperi, M. C. & Salemi, M. QuRe: software for viral quasispecies reconstruction from next-generation sequencing data. *Bioinformatics* **28**, 132–133 (2012).
91. Fullwood, M. J., Wei, C. L., Liu, E. T. & Ruan, Y. Next-generation DNA sequencing of paired-end tags (PET) for transcriptome and genome analyses. *Genome Res.* **19**, 521–532 (2009).
92. Schwartz, D. C. *et al.* Ordered restriction maps of *Saccharomyces cerevisiae* chromosomes constructed by optical mapping. *Science* **262**, 110–114 (1993).
93. Miller, J. M., Malenfant, R. M., Moore, S. S. & Coltman, D. W. Short reads, circular genome: skimming solid sequence to construct the bighorn sheep mitochondrial genome. *J. Hered.* **103**, 140–146 (2012).
94. Loman, N. J. *et al.* Performance comparison of benchtop high-throughput sequencing platforms. *Nature Biotech.* **30**, 434–439 (2012).
95. Sutton, G. G., White, O., Adams, M. D. & Kerlavage, A. R. TIGR Assembler: a new tool for assembling large shotgun sequencing projects. *Genome Sci. Technol.* **1**, 9–19 (1995).
96. Jeck, W. R. *et al.* Extending assembly of short DNA sequences to handle error. *Bioinformatics* **23**, 2942–2944 (2007).

Acknowledgements
N.N. was supported by the Agency for Science, Technology and Research (A*STAR), Singapore. M.P. was supported in part by the US National Science Foundation (grants IIS-1117247 and IIS-0844494) and by the Bill and Melinda Gates Foundation.

Competing interests statement
The authors declare no competing financial interests.

FURTHER INFORMATION
Niranjan Nagarajan's homepage: http://www.gis.a-star.edu.sg/internet/site/investigator/investigator.php?user_id=130
Mihai Pop's homepage: http://www.cbcb.umd.edu/~mpop
The Assemblathon: http://www.assemblathon.org
GAGE: http://gage.cbcb.umd.edu
Genome 10K Project: http://www.genome10k.org
i5K — ArthropodBase wiki: http://www.arthropodgenomes.org/wiki/i5K
phrap.doc: http://www.phrap.org/phredphrap/phrap.html

SUPPLEMENTARY INFORMATION
See online article: S1 (table)

ALL LINKS ARE ACTIVE IN THE ONLINE PDF

CORRECTION

In the above article, the paper cited as reference 89 was incorrect. The correct reference is:

Astrovskaya, I. *et al.* Inferring viral quasispecies spectra from 454 pyrosequencing reads. *BMC Bioinformatics* **12**, (Suppl. 6), S1 (2011).