Genome assembly reborn: recent computational challenges

Mihai Pop

Submitted: 2nd March 2009; Received (in revised form): 18th April 2009

Abstract

Research into genome assembly algorithms has experienced a resurgence due to new challenges created by the development of next generation sequencing technologies. Several genome assemblers have been published in recent years specifically targeted at the new sequence data; however, the ever-changing technological landscape leads to the need for continued research. In addition, the low cost of next generation sequencing data has led to an increased use of sequencing in new settings. For example, the new field of metagenomics relies on large-scale sequencing of entire microbial communities instead of isolate genomes, leading to new computational challenges. In this article, we outline the major algorithmic approaches for genome assembly and describe recent developments in this domain.

Keywords: *genome assembly; genome sequencing; next generation sequencing technologies*

INTRODUCTION

DNA sequencing technologies have revolutionized biology. Since the introduction of the chain termination sequencing method by Frederick Sanger in 1977 [1], the genomes of more than 800 bacteria and 100 eukaryotes have been sequenced, including the genomes of several human individuals [2–4]. Close to a trillion base pairs are currently deposited in Genbank (as of December 2008)—a central repository of genetic sequence information hosted by the NCBI—and this number is rapidly increasing. This wealth of data has resulted in numerous biological discoveries and led to a better understanding of the fundamental principles of life. The dramatic impact of sequencing as a key component of modern biological research is, at first glance, surprising due to limitations in the length of DNA fragments that can be sequenced with current technologies. Modern sequencing instruments can only 'read' DNA fragments of up to ~2000 bp (commonly just 600–1000 bp), orders of magnitude shorter than the genomes of most living organisms. Throughout the years, in fact, many thought that such limitations would prevent the sequencing of large genomes [5]. Today, however, the sequencing of bacteria (millions of base pairs in length) is done routinely and the sequencing of 1000 human genomes (3 billion bp in length, each) is considered possible within the next 3 years [6]. The apparent disconnect between the limitations of sequencing technologies and their successful application in many genome projects can be explained by the clever combination of sequencing and computation, embodied in the shotgun sequencing method proposed by Roger Staden in 1979 [7].

The shotgun process involves shearing the genome of an organism into multiple small fragments, each of which being then sequenced separately. The resulting DNA segments are combined into a reconstruction of the original genome using computer programs called genome assemblers. The assembly process is often compared to solving a jigsaw puzzle—metaphor that highlights several challenges. First, the assembly problem is complicated by genomic repeats—sections of DNA that occur in a near-identical form throughout a genome—equivalent to large stretches of sky in a jigsaw puzzle. Second, the complexity of a jigsaw

Corresponding author. Mihai Pop, Department of Computer Science, Center for Bioinformatics and Computational Biology, Biomolecular Sciences Building, Room 3120F, University of Maryland, College Park, MD 20742, USA. E-mail: mpop@umiacs.umd.edu

Mihai Pop is an assistant professor in the Department of Computer Science and the Center for Bioinformatics and Computational Biology at the University of Maryland, College Park. Tel: 301-405-7245; Fax: 301-314-1341; E-mail: mpop@umd.edu

puzzle increases dramatically with the number of pieces (a puzzle with 1000 pieces is more than twice as hard as a puzzle with 500 pieces). Similarly, the difficulty of the assembly problem is dependent on the number of reads being assembled—large genomes and/or short DNA fragments posing specific challenges. To overcome such challenges, sophisticated computational algorithms have been developed over the years, resulting in genome assemblers capable of reconstructing large mammalian genomes [8, 9]. These programs have been critical to the success of many recent genome projects including mammals [3], plants [10] and worms [11]. Despite such successes, genome assembly is far from being a solved problem. New challenges are posed by recent advances in genome sequencing technologies [12, 13], both due to the sheer size of the data that need to be processed (a single sequencing instrument can provide throughput similar to what was previously only achievable at a genome center), and due to the characteristics of the new data: shorter sequencing reads and new types of sequencing errors. Furthermore, assembly software must adapt to new applications of DNA sequencing in biological research. Perhaps the biggest new challenge is posed by the large-scale sequencing of entire microbial communities (metagenomics). The goal of this article is to summarize recent developments in genome assembly aimed at addressing the challenges posed by new sequencing technologies and by the application of sequencing beyond single genomes.

SHOTGUN SEQUENCING

Before proceeding, it is important to closer examine the shotgun-sequencing process we briefly outlined in the introduction. To recap, the process starts by randomly shearing a genome of interest into a collection of fragments. A subset of the fragments is then selected, usually those that fall within a prescribed size range, and their sequence is 'read' using a sequencing instrument, resulting in a collection of reads—DNA fragments whose sequence is known. Commonly, the length of the fragments exceeds the read length achievable by a sequencing technology, therefore only the ends of DNA fragments are being sequenced. This feature has resulted in a variant of the shotgun method, double-barreled shotgun sequencing, wherein sequence reads are generated from both ends of each DNA fragment, resulting in a collection of read pairs (mate-pairs) that are separated by a known distance (the size of the original fragment).

Shotgun sequencing can be viewed as a random sampling process, each DNA fragment originating from a random location within the genome. It can be mathematically shown [14] that a certain amount of over-sampling is necessary in order to ensure (with high probability) that every base in the genome is sampled by at least one of the reads. The amount by which a genome is over-sampled is commonly referred to as coverage—the ratio between the cumulative size of the set of reads and the size of the genome (3-fold or 3× coverage implies that the set of reads span three times as much DNA as the genome being sequenced).

SEQUENCING TECHNOLOGIES

Before discussing assembly algorithms it is important to briefly survey recent developments in genome sequencing technologies. For an in-depth description of these technologies please see [12, 13]. Furthermore, the challenges these technologies pose to assembly algorithms are summarized in Table 1. For almost 30 years, genome sequencing was primarily performed with a technology called Sanger sequencing (named after its creator Frederick Sanger). Sanger sequencing instruments can produce relatively long reads (1000–2000 bp can be routinely achieved). Mate-pair libraries can be constructed for a wide range of insert sizes (spacing between the paired reads), including 2–10 kbp (clone), 35–40 kbp (fosmid) and 50–150 kbp (BAC). A series of new sequencing technologies have emerged in recent years, starting with the introduction of the 454 pyrosequencing machine in 2005 [15], followed by Solexa/Illumina in 2006, the SOLiD technology from Applied Biosystems in 2007 and the Helicos single-molecule sequencing technology in 2008 [16]. Common to all these technologies is the high degree of parallelism in the sequencing process. Millions of DNA fragments are immobilized to a surface then sequenced simultaneously, leading to a throughput order of magnitude higher than that achievable through Sanger sequencing. This performance comes at a price: read lengths are considerably shorter, ranging from ~400 bp (454 Titanium instrument [17]) to 25–50 bp (SOLiD and Helicos). Furthermore,

Table I: Summary of main features of the new generation sequencing data and their impact on assembly software

Sequencing technology feature	Assembly challenge
Short reads	Difficulty assembling repeats
Mate-pairs absent or difficult/expensive to obtain	Difficulty assembling repeats Lack of scaffolding information
New types of errors	Need to modify existing software and/or incorporate technology-specific features in assembly software
Large amounts of data (number of reads and size of auxiliary information)	Efficiency issues Require parallel implementations or specialized hardware when applied to large genomes

The individual features do not necessarily apply to all new sequencing technologies.

while mate-pair protocols are available for these technologies, they can add substantial costs to the sequencing process—in practice most data being generated are un-mated reads. An exception is the SOLiD technology, which was designed specifically to generate mate-pair information. In addition to short read length, some of the new technologies have specific error characteristics that complicate the assembly process. The 454 technology can 'stutter' in homopolymer regions, i.e. has difficulty estimating the length of DNA stretches consisting of a single repeated nucleotide (e.g. AAAAA), while current Helicos instruments sequence each DNA fragment two or more times in order to correct for high error rates, leading to duplicated reads. All current sequencing technologies produce an estimate of the quality of the data being produced, in the form of phred quality values [18, 19]—the logarithm of the probability that a particular base in the sequence was incorrectly read. This information can be used throughout the assembly process to distinguish between sequencing errors and true differences between the reads being assembled.

GENOME ASSEMBLY

When discussing genome assembly it is important to distinguish between *de novo* approaches, aimed at reconstructing genomes that are not similar to any organisms previously sequenced, and comparative (re-sequencing) approaches that use the sequence of a closely related organism as a guide during the assembly process. Mathematically, the *de novo* genome assembly problem can be proven to be difficult, falling within a class of problems (NP-hard) for which no efficient computational solution is known [20, 21]. In contrast, comparative assembly is a much easier task—it is essentially sufficient to align the set of reads to the reference genome in order to characterize a newly sequenced organism. The two approaches are not exclusive—even if a reference genome is available, regions of the sequenced genome that differ significantly from the reference (e.g. large insertions) can only be reconstructed through *de novo* assembly. The characteristics of the sequencing technology being used also restrict the choice of assembly strategy. Short reads and the absence of mate-pairs make it difficult for *de novo* assemblers to resolve repeats, while the large number of reads generated by new generation sequencing machines leads to efficiency issues. Due to these challenges, short-read sequencing technologies (Solexa, SOLiD) have primarily been used in re-sequencing applications. *De novo* assembly of these data has largely been restricted to bacterial genomes, though an assembly of an entire human genome from Solexa reads was recently reported [22]. Irrespective of genome size, *de novo* assemblies constructed from short-read data are highly fragmented [22, 23]. This approach is, therefore, better suited for long reads or for combinations of data from multiple sequencing technologies [24].

De novo methods, however, are essential to our efforts to characterize the biological diversity in our world. Comparative approaches can only be applied to the few genomes for which reference sequences are available. In this section we will primarily focus our discussion on the main strategies for *de novo* assembly—greedy, overlap-layout-consensus and Eulerian—and conclude with a brief overview of comparative approaches.

Overlap computation

A key component of the greedy and overlap-layout-consensus (OLC) strategies is a module, called an overlapper that computes all pairwise alignments

between a set of reads. The computation of overlaps is one of the most time-intensive components of an assembly program requiring, in the worst case, time proportional to the square of the number of reads provided to the assembler (each read must be compared to all other reads, leading to $\binom{n}{2}$ operations). In most practical cases the performance is much better—running times roughly proportional to the number of reads can be achieved through simple indexing strategies. An example are indices based on exact matches of length k (k-mers)—only the reads sharing a same k-mer need to be compared when computing overlaps. Note that a quadratic behavior can still be observed in repeat regions—all reads sharing a same k-mer must be compared to each other. For this reason, many assemblers exclude from the index k-mers that appear repetitive, i.e. those contained in more reads than expected given the sequencing coverage.

Overlap computation can be easily parallelized, allowing the running time to be reduced through the use of multi-processor machines (most modern desktops contain several processors) or computational grid resources (available at most major research institutions). It is important to note that the use of short-read sequencing technologies has a significant impact on the complexity of overlap computation, in particular due to the increased number of reads generated in a short-read sequencing project—such projects generate an order of magnitude more reads than were commonly generated in Sanger-based projects.

Greedy

Greedy algorithms represent the simplest, most intuitive, solution to the assembly problem. Individual reads are joined together into contigs in an iterative fashion, starting with the reads that overlap best, and ending once no more reads or contigs can be joined. By overlap we mean that the prefix of one of the reads shares sufficient similarity with the suffix of another read (Figure 1A). The definition of overlap quality depends on implementation, commonly used measures being the length of the overlap and the level of identity (percentage of base pairs shared by the two reads) between the reads within the overlapping region. The term 'greedy' refers to the fact that the decisions made by the algorithm optimize a local objective function (in the case of assembly, the quality of the overlap between two reads), approach that may not lead to a globally optimal solution. For example, by always processing the best overlap first, a greedy assembler may misassemble repeats (Figure 1B and C).

The variant of the greedy approach outlined above was used by several of the most widely used genome assemblers for Sanger data, such as phrap, TIGR Assembler and CAP3. Recent software aimed at assembling short-read sequencing data (SSAKE, VCAKE and SHARCGS), use a different greedy strategy. An unassembled read is chosen to start a contig, which is then repeatedly extended by identifying reads that overlap the contig on its 3′ end until no more extensions are possible. The process is repeated in the 5′ direction using the reverse complement of the contig sequence. The assembly continues in an iterative fashion by scanning through the unassembled reads. The reads are considered in decreasing order of their quality, as defined either by depth of coverage (number of reads confirming a section of a given read) [25] or by using a combination of coverage, quality values and the presence of at least one perfect overlap with

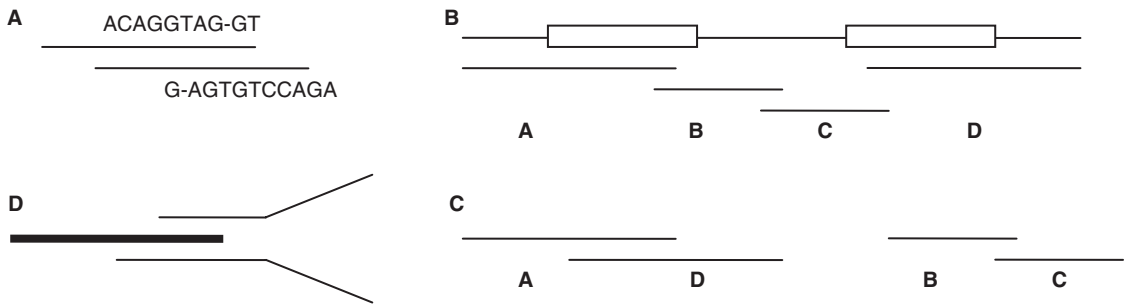

Figure 1: (**A**) Overlap between two reads—note that agreement within overlapping region need not be perfect; (**B**) Correct assembly of a genome with two repeats (boxes) using four reads A–D; (**C**) Assembly produced by the greedy approach. Reads A and D are assembled first, incorrectly, because they overlap best and (**D**) Disagreement between two reads (thin lines) that could extend a contig (thick line), indicating a potential repeat boundary. Contig extension must be terminated in order to avoid misassemblies.

another read [26]. To avoid misassemblies the extension process is terminated once conflicting information is found, i.e. two or more reads that could extend the contig; however, these reads do not overlap each other (Figure 1D).

Overlap-layout-consensus (OLC)

This strategy breaks down the assembly into three distinct steps in order to enable a global analysis of the relationships between the reads unlike the inherently localized approach taken by the greedy approaches. The first step is the same as for the greedy approach—the reads are compared to each other to construct a list of pair-wise overlaps. This information is then used to construct an overlap graph—a graph containing each read as a node, and an edge connects two nodes if an overlap was identified between the corresponding reads. During the layout stage, the overlap graph is analyzed in order to identify paths through the graph that correspond to segments of the genome being assembled. The ultimate goal is a single path that traverses each node in the overlap graph exactly once, corresponding to a reconstruction of the genome using all the sequencing reads provided as input to the assembler. Note that in the general case identifying such a path (called a Hamiltonian path) is computationally difficult, falling in the same class of NP-hard problems as the general *de novo* assembly problem. Nevertheless, the graph formulation for assembly enables several analyses not possible in the greedy setting. For example, the genome represented in the overlap graph shown in Figure 2 can be inferred to contain two copies of segment B separated by a copy of segment C (the genome 'reads' ABCBD). The greedy strategies described above would either construct a fragmented assembly that breaks at the boundary of the repeat B (a possible reconstruction is AB, C, D) or incorrectly traverse the repeat, leading to a misassembly, i.e. reconstructing a DNA segment not actually present in the genome being assembled (e.g. ABD, C). The final step in the OLC strategy is consensus computation—determining the DNA sequence that is implied by the arrangement of reads along the chosen path through the graph. This latter step is performed by allowing the reads that overlap a same base within the reconstructed genome to vote on the identity of the base, the votes being weighted by sequence quality values [27].

The layout stage, the core of the OLC strategy, merits further discussion. Practical implementations of OLC assemblers use a hierarchical approach for the layout task. First, segments of the overlap graph are identified that can be unambiguously assembled, representing segments of DNA that are definitely present in the assembled genome—generally DNA segments that do not contain repeats, or that are entirely contained within a single copy of a repeat. For example Celera Assembler defines the concept of a unitig (uniquely assemblable contig) that can be constructed by following overlap edges until encountering a 'fork' in the graph. By fork we mean a read A that overlaps two other reads, B and C; however, B and C do not overlap each other. Such a situation often represents the boundary between a repeat and the genomic regions adjacent to the copies of this repeat throughout the genome; however, forks can also be caused by sequencing errors (e.g. the overlap between B and C is obscured by errors in one or both of the reads). Forks caused by errors can be resolved through simple heuristics,

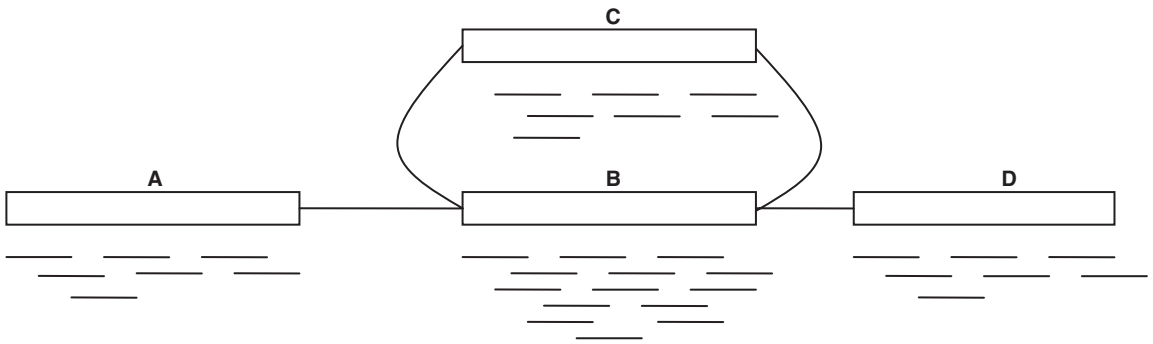

Figure 2: Overlap graph of a genome containing a two-copy repeat (B). Note the increased depth of coverage within the repeat. The correct reconstruction of this genome spells the sequence ABCBD, while conservative assembly approaches would lead to a fragmented reconstruction.

e.g. by removing 'spurs'—single reads that disagree with the bulk of the reads within a region of the assembly graph [8, 28] —or by adding to the graph the overlaps that were 'hidden' by sequencing errors [29]. Unitigs can also be constructed with the help of mate-pair information—Arachne seeds the assembly process with groups of mate-pairs that overlap at both ends (paired pairs) [29].

Unitig construction is intentionally conservative and rarely results in misassemblies with the exception of potentially collapsing multiple copies of a repeat into a single unitig. The resulting set of unitigs is, however, highly fragmented as unitigs cannot cross repeat boundaries. A unitig-only assembly is sufficient for some applications, e.g. for Chip-Seq experiments [30] or in transcriptome sequencing [31]. In such cases unitigs can be obtained with standalone assemblers such as Minimus [32] or newbler (assembler provided with the 454 sequencing instrument), or by examining the internal datastructures of Celera Assembler or Arachne.

For the assembly of whole genomes, the unitig reconstruction is just a first step in the assembly process and is followed by more sophisticated analyses of the assembly graph aimed at reconstructing larger contiguous segments of the genome being assembled. For example, Celera Assembler and Arachne use mate-pair information to identify sets of unitigs that could be merged into larger structures. Further, algorithms based on network flow analysis have been proposed to identify paths through the assembly graph that accurately account for the copy number of repeats [21, 33], leading to correct reconstructions of larger segments of the genome.

The OLC strategy is arguably the most successful assembly strategy in a practical setting. The majority of the large genomes sequenced in recent years through shotgun sequencing has been assembled with either Celera Assembler or Arachne, and the rapid adoption of the 454 sequencing technology is in part due to the performance of the newbler assembler. The OLC approach was also successful for the assembly of data with very short read lengths (e.g. Solexa)—the OLC assembler Edena was shown to outperform several greedy assemblers (SSAKE, SHARCGS), as well an Eulerian path assembler (Velvet) [28]. In part, this success is due to the inherent flexibility of the OLC strategy. The approach is inherently modular and the overlap graph structure lends itself to multiple types of analyses. It is important to note that most OLC assemblers allow users to extract graph information, providing the opportunity for additional downstream analyses (e.g. this information could guide the design of finishing experiments [34].

Eulerian path

The Eulerian path strategy was inspired by early work on sequencing by hybridization (SBH) [35]. The output of an SBH experiment is a list of all oligomers of a given length k present in the genome being sequenced. This information is also referred to as the k-mer spectrum of a genome (Figure 3A) and is, conceptually, more valuable than the information produced by a shotgun sequencing experiment. In fact, the k-mer spectrum can be viewed as the output of a perfect shotgun experiment where the reads all have equal length k and perfectly sample the genome, with a read starting at every single base. The SBH technology was ultimately not successful due to technical challenges, partly because it is simply too difficult to experimentally interrogate every single k-mer in a genome with the exception of small values of k (corresponding to very short sequencing reads). However, the theoretical analysis of the problem of assembling a genome from SBH data has resulted in a new strategy for assembling a genome. This Eulerian path

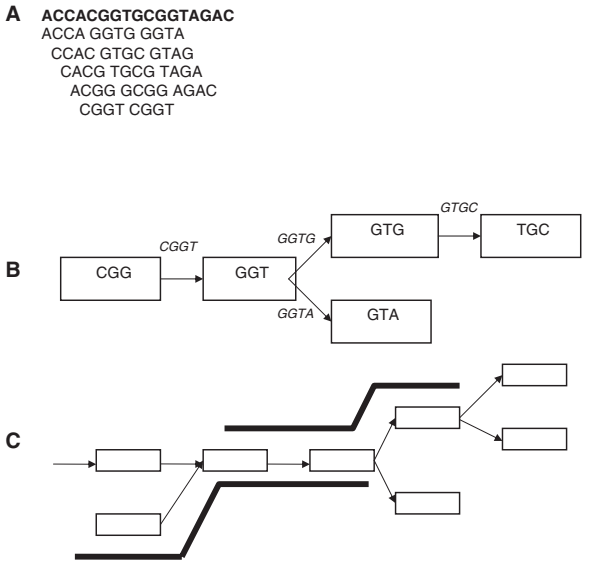

Figure 3: (**A**) k-mer spectrum of a DNA string (bold) for $k = 4$; (**B**) Section of the corresponding deBruijn graph. The edges are labeled with the corresponding k-mer and (**C**) Overlap between two reads (bold) that can be inferred from the corresponding paths through the deBruijn graph.

approach starts by breaking up the set of reads into their k-mer spectrum, effectively reconstructing the information that would be obtained in an SBH experiment. In fact, the data extracted from reads also carries additional information—the abundance (coverage) of each k-mer, information that can be useful in resolving repeats. The resulting k-mer spectrum is then used to construct a deBruijn graph—a graph containing as nodes the $k-1$ length prefixes and suffixes of the original k-mers, two nodes being linked by an edge if the adjacent $k-1$-mers have an exact overlap of length $k-2$ (Figure 3B). In effect each k-mer identified within the set of reads is converted into an edge of the deBruijn graph, and the assembly problem becomes equivalent to finding a path through this graph that uses every edge in the graph (we know each k-mer is present in the genome, therefore the corresponding edge must be included in the reconstruction). Such a path is called a Eulerian path, hence the name for this assembly strategy.

Intuitively, the Eulerian approach offers several advantages over the OLC strategy. First of all, pairwise overlaps between reads are never explicitly computed, hence no expensive overlap step is necessary, rather overlaps are implicitly represented in the deBruijn graph by paths that traverse from one read to its neighbor (Figure 3C). Furthermore, efficient algorithms exist for finding a Eulerian path in a graph, in contrast to the OLC approach that leads to the difficult task of finding a Hamiltonian path. This intuition, however, ignores the fact that, in general, an exponential number of distinct Eulerian paths can be found in a graph, corresponding to the many different ways a genome can be rearranged around its repeats. Ultimately, the task of an assembler is to find just one of the possible paths, corresponding to the correct reconstruction of the genome. Adding additional constraints to the Eulerian path approach in order to guide the algorithm toward this one correct path, leads to a considerably harder computational problem [21]. In fact, simple transformations of the overlap graph from the OLC paradigm leads to a graph structure (string graph) that is functionally equivalent to the deBruijn graph of a genome [33], hinting at the equivalence between these assembly paradigms.

As described above, the construction of the deBruijn graph leads to a loss of information—chopping up the reads into a set of k-mers results in a loss of long-range connectivity information implied by each read. This information is critical in reducing the ambiguity in the graph structure caused by short repeats. In order to incorporate read information in the assembly process, Pevzner *et al.* [36] formulated a new variant of the Eulerian path problem called the Eulerian superpath problem—the task of finding a Eulerian path through the graph that is constructed from sub-paths corresponding to individual reads provided as input to the assembler.

The Eulerian strategy has been proposed as an alternative to OLC for the assembly of Sanger data and was implemented in the Euler series of assemblers [36–40]; however, was not widely adopted, in part due to its sensitivity to sequencing errors. Errors lead to the creation of 'new' k-mers and dramatically increase the size and complexity of the resulting deBruijn graph, leading to the need for sophisticated error correction algorithms [36, 40]. New sequencing technologies, in particular those generating short reads, are a better target for the Eulerian strategy. These technologies generate high depths of coverage in reads that are roughly equal in length, effectively generating the same information that would be produced by an SBH experiment. Two recently developed assemblers specifically targeted at short-read sequence data use the Eulerian strategy: Velvet [41], and Allpaths [42].

Scaffolding

None of the assembly strategies described above can completely reconstruct a genome from read data alone, the output of most assemblers consisting of a (often large) collection of independent contigs. Other sources of information can be used to determine the relative placement of these contigs along a genome in a process called scaffolding. The output of this process consists of a series of scaffolds—groups of contigs whose relative placement is known even though the DNA sequence of the genomic regions connecting adjacent contigs is unknown. Most commonly, scaffolding relies on mate-pair information. Two contigs can be inferred to be adjacent in the genome if one end of a mate-pair is assembled within the first contig, and the other end is assembled within the second contig. In practice, two or more such links are required between two contigs in order to reduce the impact of experimental errors. Like *de novo* assembly, scaffolding can be shown to be computationally difficult [43]; however, simple heuristics perform well in practice. All modern

assemblers, irrespective of the underlying assembly paradigm, contain a scaffolding module: Euler [37], Arachne [29], Celera Assembler [8], Velvet [41]. Stand-alone scaffolders are also available, such as Bambus [44] allowing mate-pair information to be added to virtually any assembler.

Scaffolding algorithms often follow a greedy approach, starting with the most reliable information then iteratively incorporating more data as long as the new information does not conflict with the already constructed scaffold. In Bambus [44] the order in which mate-pairs are processed is specified by the user either by library (e.g. process clone libraries first, followed by fosmids, then BAC-ends), or by the number of links connecting adjacent contigs. In Celera Assembler [8], unique contigs that are well connected to the rest of the assembly (called rocks) are processed first, followed by contigs that are not only connected by one mate pair but also overlap an adjacent contigs (stones), followed by unconnected contigs that can be used to tile across gaps in the assembly (pebbles). Velvet [41] starts the scaffolding process with contigs longer than the mate-pair size, using the deBruijn graph information together with mate-pairs to 'walk' across the gap between these contigs. Note that in the Eulerian approach, mate-pairs define paths through the graph that need to be traversed in a reconstruction of a genome, thus mate-pair information can be processed with the same algorithms used to solve the Eulerian super-path problem [37].

Scaffolding information can also be obtained from whole-genome mapping data. In particular, optical mapping [45] has been successfully used in this context. In brief, optical mapping can determine the approximate location of restriction enzyme cuts along a genome, thereby generating an ordered list of restriction fragment lengths along the genome. Such information can be used to identify the location of assembled contigs within the genome, leading to a single genome-wide scaffold. The scaffolding process, implemented in the program SOMA [46], involves constructing *in silico* restriction maps from a set of contigs, then mapping these contigs along a genome-wide optical map. The initial application of this approach to bacterial genomes has resulted in scaffolds that cover up to 80–90% of the entire genome sequence [46].

Mate-pair and optical mapping data provide complementary information to the scaffolding process. Mate-pairs can offer high-resolution but inherently local information while optical maps generate a global, low-resolution view of the genome. These technologies can be effectively combined, for example, by using an optical map to anchor a set of large contigs along the genome, then using mate-pair information to complete the gaps within the scaffold. Note that both mate-pair and mapping-based scaffolding approaches have difficulties scaffolding short contigs and may, therefore, be difficult to apply to fragmented assemblies generated from short-read sequencing data.

Comparative assembly

Often the genome being sequenced is closely related to a genome that has been previously sequenced. This is the case, for example, when studying genomic variation within a population, e.g. in human resequencing experiments, or when sequencing multiple strains of a same bacterium. The available reference sequence can be used to guide the assembly of a genome using a process called comparative, or templated assembly. Briefly, the reads are mapped to the reference genome and their placement is used to infer the structure of the genome being sequenced (target genome). In this process care must be taken to avoid obscuring differences between the two genomes. For example, the comparative assembler AMOScmp [47] identifies locations in the reference where the alignment of multiple reads 'breaks' and flags these regions as the location of possible insertions or deletions between the two genomes. In the context of resequencing experiments using short-read sequence data, the program Maq [48] uses a probabilistic framework to assign a quality value to each read alignment, then uses this information together with sequence quality data in order to characterize single nucleotide polymorphisms (SNPs) between the target genome and the reference. Furthermore Maq assumes that the target genome is haploid, thus each SNP may represent one of three possible genotypes (aa, ab, bb).

Finally, the comparative approach can also be applied when the reference is a protein sequence. In this case the goal is to reconstruct an individual gene rather than organism. Such an approach is implemented in the assembler ABBA [49]. Here the reads are translated in all six reading frames then aligned to the sequence of a protein of interest and the alignment information is used to guide the assembly process. Since protein sequences are

conserved across longer evolutionary distances than DNA, protein-guided assembly can be successful even if no closely related reference sequences are available. Further, this approach is effective even for very short reads (25 bp/eight amino acids) as long repeats are unlikely to occur within the span of a single gene.

ASSEMBLY VALIDATION

No genome assembler is perfect. Assembly errors can occur either due to limitations of the assembly algorithm, or due to incomplete or incorrect information provided to the assembler. The detection of assembly errors is important both during the assembly process, in which case such errors can be corrected [9, 50], but also post-assembly, either during genome finishing or to inform downstream analyses of the assembly data. Errors can often be detected by identifying several types of inconsistencies in the assembled data. Collapsed repeats, for example, lead to regions within the assembly that have unusually deep coverage. Such regions can be identified through statistical approaches: Kim *et al.* [51] use an ad hoc probability distribution based on *k*-mer frequencies to estimate the likelihood that an individual read contributes to a misassembly, while Celera Assembler [8] and Velvet [41] use Poisson statistics to estimate the likelihood that a particular genomic region represents a collapsed repeat. Large numbers of disagreements between reads within the assembly can also be caused by collapsed repeats due to differences between the sequence of different repeat copies. To identify such regions the DNP framework of Tammi *et al.* [52] estimates the likelihood that two polymorphisms occur nearby each other using sequence quality values and Poisson statistics. When identified, such errors can be resolved either by splitting apart the set of reads into multiple repeat copies—approach taken by DNPtrapper [53]—or by shuffling reads around the genome in order to improve overall quality of the consensus sequence—as implemented in Euler-AIR [54].

Mate-pairs provide a powerful tool for identifying large-scale misassemblies. In a correct assembly, the tiling of reads should be consistent with the constraints imposed by mate-pairs (relative orientation and spacing of paired reads). Deviations from this ideal, especially when confirmed by multiple mate-pairs, often indicate the presence of errors. The Tampa program [55] uses a geometric approach to identify assembly errors using mate-pair data, that can also categorize the type of error identified (insertion, deletion, inversion or transposition). The C/E statistic of Zimin [56] estimates the likelihood that a cluster of mate-pairs indicates an insertion or deletion within the assembly. Visual tools for validating assemblies with the help of mate-pair information are available in a variety of packages including consed [57], BACCardI [58] and Hawkeye [59].

Finally, whole genome mapping data can be effectively used to assess the quality of an assembly, by comparing an *in silico* map constructed from the set of contigs to the experimentally derived map. In the context of optical restriction maps, when using the scaffolder SOMA [46], contigs that do not have a good match to the optical map can be inferred to represent misassemblies. Map-based validation has been used in multiple genome projects including, e.g. human [60], drosophila [8], *Plasmodium falciparum* [61], to name just a few. Note, however, that contigs must be large enough to contain multiple markers in order to be accurately compared to a genome map, limiting the effectiveness of this approach in the context of short-read sequencing data.

Due to inherent experimental variability in the sequencing process, validation methods relying on just one source of information can result in many false positives (regions incorrectly tagged as misassemblies). This problem can be alleviated integrating multiple sources of information in the validation process. Two recently developed software packages provide a framework for achieving such integration. In Amosvalidate [62], the output of multiple validation modules is reported as a series of features along the assembly and regions that are densely populated by such features are flagged for further inspection. The output of Amosvalidate is integrated with the viewer Hawkeye allowing the visual inspection of the flagged regions. Choi *et al.* [63] propose the integration of multiple validation methods through machine-learning techniques. They use data with known misassembly characteristics (e.g. assemblies of a genome for which a reference is available) to train the weights associated with individual measures of assembly correctness and show that this approach leads to a better prediction of misassemblies than methods relying on just one source of information.

INFORMATION INTEGRATION

The recent availability of multiple sequencing technologies provides the opportunity for a hybrid assembly approach wherein different sources of data are combined to take advantage of their complementary strengths. Directly integrating data from different technologies into an assembler can be difficult due to the different characteristics of the data. For example, overlap computation can be difficult when mixing reads of substantially different lengths, e.g. when combining Solexa with Sanger sequencing data. The Eulerian strategy is perhaps the best suited for hybrid assembly as read length becomes immaterial once a *k*-mer spectrum is constructed; however, none of the existing Eulerian assemblers (Euler-SR [39], Euler-USR [40], Velvet [41], Allpaths [42]) allow such functionality. In the context of the OLC paradigm, proposed approaches for integrating different types of sequencing data have relied on the use of multiple genome assemblers. Goldberg *et al.* [64] combine Sanger and 454 data by first assembling the 454 data with the newbler assembler, then chopping the resulting contigs into a set of Sanger-like reads which are then assembled together with the Sanger data using Celera Assembler. A similar approach is used by Reinhardt *et al.* [24] to combine Solexa and 454 data. In this case the Solexa data were first assembled with VCAKE, then entire contigs were combined with 454 data using the newbler assembler. Due to the small size of contigs generated by VCAKE, no additional shearing of these contigs was necessary prior to assembly with newbler. Sundquist *et al.* [65] propose to use 454 sequencing to assemble large genomes by integrating information provided by a hybrid sequencing strategy. Their SHRAP strategy involves the construction of a BAC (or other large insert) library from a genome, then sequencing each individual clone through 454 sequencing. They show that the resulting data can be used to simultaneously build an assembly and construct a clone map of the genome (the latter step is an expensive component of traditional sequencing approaches).

METAGENOMICS/MIXTURES OF ORGANISMS

New assembly challenges are created by the sequencing of mixtures of organisms, primarily through metagenomic projects (sequencing of entire microbial communities). The assembly software described in this article was designed specifically for the assembly of isolate genomes, i.e. the expected output from these assemblers is a linear (or in the case of most bacteria, circular) sequence. These assemblers can be confused by two main characteristics of metagenomic data: (i) uneven representation of the organisms within a sample and (ii) polymorphisms between closely related members of an environment. As described above, many assemblers use depth of coverage statistics to identify repeats. In metagenomic data, due to uneven coverage, these assemblers would incorrectly label as repeats the most abundant members of a community, and avoid assembling these genomes to prevent misassemblies—phenomenon observed, for example, when Celera Assembler was used to assemble microbial communities from the ocean [66]. Differences between very closely related organisms (e.g. prophages inserted at different locations within individuals from a same strain, or genomic rearrangements) make it impossible to construct a single consensus sequence. Instead, most assemblers construct unrelated contigs for each section of the genome that is consistent across multiple individuals in the sample, resulting in a fragmented reconstruction and obscuring the population structure within the environment [67]. Assemblers designed for large eukaryotic projects allow for limited polymorphisms between the two homologous chromosomes being assembled; however, they have difficulty assembling highly polymorphic genomes [68]. Initial metagenomic projects have primarily used traditional assemblers—phrap was used to assemble a microbial community from acid mine drainage [69], and Celera Assembler was used to assemble ocean [66] and human fecal communities [70]. Specialized software was used in such studies to infer community structure from the assembly data [71, 72]. Mavromatis *et al.* [73] performed simulations comparing the performance on metagenomic data of several assemblers (phrap, Arachne and Jazz—an assembler used in-house at the Joint Genome Institute) and concluded that Arachne provided the best trade-off between contiguity and number of errors (in this case errors refer to situations when reads from distinct organisms are co-assembled). A promising approach to the assembly of mixtures, in particular in the case of polyploid organisms, is provided by Allpaths [42]. This assembler produces as output a graph structure representing the ambiguity in the structure of the

reconstructed genome, and this graph can be used to identify and analyze regions of polymorphism.

CONCLUSIONS

We hope our article provides the reader with a suitable overview of current challenges in genome assembly research and we would like to conclude our review with a few comments. First, as sequencing technologies continue to evolve, assembly algorithms and software will have to keep up with this ever-changing field. Research efforts must continue to better establish how data characteristics affect the assembly process. Chaisson *et al.* [40] have, for example, attempted to quantify the effect of read length on the ability to assemble a genome, revealing a surprising result—increases in read lengths beyond ~35–60 bp do not necessarily yield significant improvements in the quality of assembly when mate-pairs are available. Their results are obtained in a very limited setting: they focused on small genomes (bacteria and yeast) and used mate-pairs—information that can add significant costs to the sequencing process and is often unavailable. More studies are needed to determine whether these results hold for large repeat-rich organisms, and many more questions remain to be answered, e.g. is assembly possible with very long (10–100 kb) reads with high error rates? Second, research must continue on automated (both computational and experimental) approaches for validating genome assembly. The wealth of data being generated by the new sequencing technologies can no longer be validated 'by hand', and we will soon see a significant decrease in the role assembly viewers (such as consed or Hawkeye) play in genomic research. Third, it is likely that multiple sequencing technologies will co-exist in the future technological landscape, leading to the need for new approaches for combining multiple types of data in the assembly process, including non-sequence data such as that obtained through high throughput mapping technologies. Fourth, the assembly of mixtures of organisms will become routine in the near future, perhaps surpassing in volume the sequencing of isolated genomes. Software tools need to be developed that are specifically targeted at such mixed samples. New ways to represent assembly data, such as the graphs produced by Allpaths, must also be developed as well as analysis tools able to process these new types of data. Finally, the development of new sequencing technologies has resulted in the development of many new assemblers even though the algorithms employed by these assemblers closely resemble algorithms already implemented in the assemblers developed for Sanger data. This trend is impractical as new sequencing technologies continue to be developed, rather future genome assemblers will have to be able to adapt to the changing technological landscape with few modifications.

Key Points

- Next generation sequencing technologies are creating new challenges for genome assembly software.
- Additional assembly challenges are posed by efforts to sequence mixtures of organisms (metagenomics).
- Multiple genome assemblers have been created in recent years to address such challenges but more research is necessary due to an ever-changing technological landscape and new applications of sequencing technologies.

Acknowledgements

The author would like to thank the anonymous reviewers for their help in improving the manuscript.

FUNDING

NSF under grant IIS-0812111 and by the NIH under grant 1-R01-HG004885-01.

References

1. Sanger F, Nicklen S, Coulson AR. DNA sequencing with chain-terminating inhibitors. *Proc Natl Acad Sci USA* 1977; **74**:5463–67.
2. Wang J, Wang W, Li R, *et al.* The diploid genome sequence of an Asian individual. *Nature* 2008;**456**:60–65.
3. Levy S, Sutton G, Ng PC, *et al.* The diploid genome sequence of an individual human. *PLoS Biol* 2007;**5**:e254.
4. Wheeler DA, Srinivasan M, Egholm M, *et al.* The complete genome of an individual by massively parallel DNA sequencing. *Nature* 2008;**452**:872–6.
5. Green P. Against a whole-genome shotgun. *Genome Res* 1997;**7**:410–7.
6. Siva N. 1000 Genomes project. *Nat Biotechnol* 2008;**26**:256.
7. Staden R. A strategy of DNA sequencing employing computer programs. *Nucleic Acids Res* 1979;**6**:2601–10.
8. Myers EW, Sutton GG, Delcher AL, *et al.* A whole-genome assembly of Drosophila. *Science* 2000;**287**:2196–204.
9. Jaffe DB, Butler J, Gnerre S, *et al.* Whole-genome sequence assembly for Mammalian genomes: arachne 2. *Genome Res* 2003;**13**:91–6.
10. Ming R, Hou S, Feng Y, *et al.* The draft genome of the transgenic tropical fruit tree papaya (Carica papaya Linnaeus). *Nature* 2008;**452**:991–6.

11. Abad P, Gouzy J, Aury JM, *et al*. Genome sequence of the metazoan plant-parasitic nematode. *Meloidogyne incognita*. *Nat Biotechnol* 2008;**26**:909–15.
12. Pop M, Salzberg SL. Bioinformatics challenges of new sequencing technology. *Trends Genet* 2008;**24**:142–9.
13. Mardis ER. The impact of next-generation sequencing technology on genetics. *Trends Genet* 2008;**24**:133–41.
14. Lander ES, Waterman MS. Genomic mapping by fingerprinting random clones: a mathematical analysis. *Genomics* 1988;**2**:231–9.
15. Margulies M, Egholm M, Altman WE, *et al*. Genome sequencing in microfabricated high-density picolitre reactors. *Nature* 2005;**437**:376–80.
16. Harris TD, Buzby PR, Babcock H, *et al*. Single-molecule DNA sequencing of a viral genome. *Science* 2008;**320**:106–9.
17. Zehr JP, Bench SR, Carter BJ, *et al*. Globally distributed uncultivated oceanic N2-fixing cyanobacteria lack oxygenic photosystem II. *Science* 2008;**322**:1110–2.
18. Ewing B, Green P. Base-calling of automated sequencer traces using phred. II. Error probabilities. *Genome Res* 1998;**8**:186–94.
19. Ewing B, Hillier L, Wendl MC, *et al*. Base-calling of automated sequencer traces using phred. I. Accuracy assessment. *Genome Res* 1998;**8**:175–85.
20. Myers EW. Toward Simplifying and Accurately Formulating Fragment Assembly. *J. Comp. Bio* 1995;**2**: 275–90.
21. Medvedev P, Georgiou K, Myers EW, *et al*. 'Computability and Equivalence of Models for Sequence Assembly'. *Workshop on Algorithms in Bioinformatics (WABI 2007)*. Philadelphia, PA: Springer, 2007.
22. Simpson JT, Wong K, Jackman SD, *et al*. ABySS: a parallel assembler for short read sequence data. *Genome Res* 2009, doi:10.1101/gr.089532.108 (epub ahead of print).
23. Farrer RA, Kemen E, Jones JD, *et al*. *De novo* assembly of the *Pseudomonas syringae* pv. syringae B728a genome using Illumina/Solexa short sequence reads. *FEMS Microbiol Lett* 2009;**291**:103–11.
24. 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.
25. Jeck WR, Reinhardt JA, Baltrus DA, *et al*. Extending assembly of short DNA sequences to handle error. *Bioinformatics* 2007;**23**:2942–4.
26. Dohm JC, Lottaz C, Borodina T, *et al*. SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. *Genome Res* 2007;**17**: 1697–1706.
27. Churchill GA, Waterman MS. The accuracy of DNA sequences: estimating sequence quality. *Genomics* 1992;**14**: 89–98.
28. Hernandez D, Francois P, Farinelli L, *et al*. *De novo* bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome Res* 2008;**18**:80209.
29. Batzoglou S, Jaffe DB, Stanley K, *et al*. ARACHNE: a whole-genome shotgun assembler. *Genome Res* 2002;**12**: 177–89.
30. Johnson DS, Mortazavi A, Myers RM, *et al*. Genome-wide mapping of *in vivo* protein–DNA interactions. *Science* 2007; **316**:1497–502.
31. Mortazavi A, Williams BA, McCue K, *et al*. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods* 2008;**5**:621–8.
32. Sommer DD, Delcher AL, Salzberg SL, *et al*. Minimus: a fast, lightweight genome assembler. *BMC Bioinformatics* 2007;**8**:64.
33. Myers EW. The fragment assembly string graph. *Bioinformatics* 2005;**21**(Suppl 2):ii79–85.
34. Mulyukov Z, Pevzner PA. EULER-PCR: finishing experiments for repeat resolution. *Pac Symp Biocomput* 2002; 199–210.
35. Idury RM, Waterman MS. A new algorithm for DNA sequence assembly. *J. Comp. Bio* 1995;**2**:291–306.
36. Pevzner PA, Tang H, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA* 2001;**98**:9748–53.
37. Pevzner PA, Tang H. Fragment assembly with double-barreled data. *Bioinformatics* 2001;**17**(Suppl 1):S225–33.
38. Chaisson M, Pevzner P, Tang H. Fragment assembly with short reads. *Bioinformatics* 2004;**20**:206774.
39. Chaisson M, Pevzner P. Short read fragment assembly of bacterial genomes. *Genome Research* 2007;**18**: 324–330.
40. 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.
41. Zerbino DR, Birney E. Velvet: algorithms for *de novo* short read assembly using de Bruijn graphs. *Genome Res* 2008;**18**:821–9.
42. Butler J, MacCallum I, Kleber M, *et al*. ALLPATHS: *de novo* assembly of whole-genome shotgun microreads. *Genome Res* 2008;**18**:810–20.
43. Huson DH, Reinert K, Myers E. 'The Greedy Path-Merging Algorithm for Sequence Assembly'. In: Lengauer T, Sankoff D, Istrail S, *et al*. (eds). *Proceedings of the Fifth Annual International Conference on Computational Biology (RECOMB)*, 2001, pp. 157–163. Association for Computing Machinery, Montreal, Canada.
44. Pop M, Kosack DS, Salzberg SL. Hierarchical scaffolding with Bambus. *Genome Res* 2004;**14**:149–59.
45. Samad A, Huff EF, Cai W, *et al*. Optical mapping: a novel, single-molecule approach to genomic analysis. *Genome Res* 1995;**5**:1–4.
46. Nagarajan N, Read TD, Pop M. Scaffolding and validation of bacterial genome assemblies using optical restriction maps. *Bioinformatics* 2008;**24**:1229–35.
47. Pop M, Phillippy A, Delcher AL, *et al*. Comparative genome assembly. *Brief Bioinform* 2004;**5**:237–48.
48. Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res* 2008;**18**:1851–8.
49. Salzberg SL, Sommer DD, Puiu D, *et al*. Gene-boosted assembly of a novel bacterial genome from very short reads. *PLoS Comput Biol* 2008;**4**:e1000186.
50. Huang X, Madan A. CAP3: A DNA sequence assembly program. *Genome Res* 1999;**9**:868–77.
51. Kim S, Liao L, Tomb JF. 'A probabilistic approach to sequence assembly validation'. In: Zaki MJ, Toivonen H, Wang JT (eds). *Workshop on Data Mining in Bioinformatics*, 2001, pp. 38–43. Association for Computing Machinery, San Francisco, CA.

52. Tammi MT, Arner E, Britton T, *et al*. Separation of nearly identical repeats in shotgun assemblies using defined nucleotide positions, DNPs. *Bioinformatics* 2002;**18**:379–88.
53. Arner E, Tammi MT, Tran AN, *et al*. DNPTrapper: an assembly editing tool for finishing and analysis of complex repeat regions. *BMC Bioinformatics* 2006;**7**:155.
54. Zhi D, Keich U, Pevzner P, *et al*. Correcting base-assignment errors in repeat regions of shotgun assembly. *IEEE/ACM Trans Comput Biol Bioinform* 2007;**4**:54–64.
55. Dew IM, Walenz B, Sutton G. A tool for analyzing mate pairs in assemblies (TAMPA). *J Comput Biol* 2005;**12**: 497–513.
56. Zimin AV, Smith DR, Sutton G, *et al*. Assembly reconciliation. *Bioinformatics* 2008;**24**:42–5.
57. Gordon D, Abajian C, Green P. Consed: A graphical tool for sequence finishing. *Genome Res* 1998;**8**:195–202.
58. Bartels D, Kespohl S, Albaum S, *et al*. BACCardI–a tool for the validation of genomic assemblies, assisting genome finishing and intergenome comparison. *Bioinformatics* 2005; **21**:853–9.
59. Schatz MC, Phillippy AM, Shneiderman B, *et al*. Hawkeye: an interactive visual analytics tool for genome assemblies. *Genome Biol* 2007;**8**:R34.
60. Venter JC, Adams MD, Myers EW, *et al*. The sequence of the human genome. *Science* 2001;**291**:1304–51.
61. Lai Z, Jing J, Aston C, *et al*. A shotgun optical map of the entire *Plasmodium falciparum* genome. *Nat Genet* 1999; **23**:309–13.
62. Phillippy AM, Schatz MC, Pop M. Genome assembly forensics: finding the elusive mis-assembly. *Genome Biol* 2008;**9**:R55.
63. Choi JH, Kim S, Tang H, *et al*. A machine-learning approach to combined evidence validation of genome assemblies. *Bioinformatics* 2008;**24**:744–50.
64. Goldberg SM, Johnson J, Busam D, *et al*. A Sanger/pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes. *Proc Natl Acad Sci USA* 2006;**103**:11240–5.
65. Sundquist A, Ronaghi M, Tang H, *et al*. Whole-genome sequencing and assembly with high-throughput, short-read technologies. *PLoS ONE* 2007;**2**:e484.
66. Venter JC, Remington K, Heidelberg JF, *et al*. Environmental genome shotgun sequencing of the Sargasso Sea. *Science* 2004;**304**:66–74.
67. Simmons SL, Dibartolo G, Denef VJ, *et al*. Population genomic analysis of strain variation in Leptospirillum group II bacteria involved in acid mine drainage formation. *PLoS Biol* 2008;**6**:e177.
68. Vinson JP, Jaffe DB, O'Neill K, *et al*. Assembly of polymorphic genomes: algorithms and application to Ciona savignyi. *Genome Res* 2005;**15**:1127–35.
69. Tyson GW, Chapman J, Hugenholtz P, *et al*. Community structure and metabolism through reconstruction of microbial genomes from the environment. *Nature* 2004; **428**:37–43.
70. Gill SR, Pop M, Deboy RT, *et al*. Metagenomic analysis of the human distal gut microbiome. *Science* 2006;**312**: 1355–9.
71. Rusch DB, Halpern AL, Sutton G, *et al*. The Sorcerer II global ocean sampling expedition: Northwest Atlantic through Eastern Tropical Pacific. *PLoS Biol* 2007;**5**:e77.
72. Eppley JM, Tyson GW, Getz WM, *et al*. Strainer: software for analysis of population variation in community genomic datasets. *BMC Bioinformatics* 2007;**8**:398.
73. Mavromatis K, Ivanova N, Barry K, *et al*. Use of simulated data sets to evaluate the fidelity of metagenomic processing methods. *Nat Methods* 2007;**4**:495–500.