 doi:10.1093/bfgp/elr037

Bioinformatics tools and databases for analysis of next-generation sequence data

Hong C. Lee, Kaitao Lai, Michał Tadeusz Lorenc, Michael Imelfort, Chris Duran and David Edwards

Advance Access publication date 19 December 2011

Abstract

Genome sequencing has been revolutionized by next-generation technologies, which can rapidly produce vast quantities of data at relatively low cost. With data production now no longer being limited, there is a huge challenge to analyse the data flood and interpret biological meaning. Bioinformatics scientists have risen to the challenge and a large number of software tools and databases have been produced and these continue to evolve with this rapidly advancing field. Here, we outline some of the tools and databases commonly used for the analysis of next-generation sequence data with comment on their utility.

Keywords: *De novo sequencing; sequence assembly; short reads; second-generation sequencing; next-generation sequencing; sequence alignment; visualization*

GENOME ASSEMBLY ALGORITHMS

One of the main tasks in next-generation sequence analysis is *de novo* genome assembly [1]. While early assemblers could only manage to assemble small bacterial genomes, improvements in data quality and quantity, combined with more advanced assembly algorithms and computational hardware have allowed the assembly of more complex eukaryotic genomes [2, 3]. During the first years of next-generation sequencing, several assembly algorithms were developed, some of which have kept pace with developments in data production and algorithmic enhancements, while others have fallen by the wayside.

While it is relatively easy to process data with assembly software and output some assembled sequences, the assembly process is relatively complex and a good understanding of the different approaches and associated parameters is required to produce consistent and high-quality assemblies. There are a large number of reviews and detailed manuscripts that describe assembly methods in detail [2, 4, 5] and this information will only be summarized here. One challenge of genome sequencing lies in the fact that only a small portion of the genome encodes genes, and that these genes are surrounded by repetitive DNA sequences that are often difficult to assemble. Repeats cause ambiguity during fragment assembly and so pose the greatest challenge when assembling genomic data. The most robust method to overcome the problems of repeats is to increase

Corresponding author. Dr David Edwards, School of Agriculture and Food Sciences, University of Queensland, Brisbane, QLD 4072, Australia. Phone: +61 (0)7 3346 7084; Fax: +61 (0) 7 3365 1176; E-mail: Dave.Edwards@uq.edu.au

Hong Ching Lee research interests include developing tools and algorithms for genomic and protein analysis. My current focus is on the association of markers with important agronomic traits in plant species.

Kaitao Lai research interests involve analysing gene duplication, and identifying genetic variations, including genetic diversity analysis and the association of markers with important agronomic traits in *Triticum aestivum*.

Michał Tadeusz Lorenc research interests involve the discovery, annotation and visualization of molecular genetic markers using next-generation sequence data, with a focus on crop plants species.

Michael Imelfort research interests include developing tools and algorithms for DNA sequence manipulation. Areas include sequence mapping and annotation as well as genomic and meta-genomic assembly, including *de novo* assembly.

Chris Duran research interests include the infrastructural challenges in management of next-generation data, the discovery and annotation of molecular genetic markers, with a focus on plant species.

David Edwards research interests include applied agricultural biotechnology, the structure and expression of plant genomes, the discovery and application of molecular genetic markers and applied bioinformatics, with a focus on crop plants.

the read length to a point that every repeat is spanned by at least one read. However, most second-generation sequencing technologies compromise read length to deliver an increased number of reads, and while third-generation technologies offer the promise of longer reads, these are yet to be applied for routine sequencing of complex genomes.

Modifications of second-generation sequencing methods attempt to overcome the problem of short reads by producing read pairs. If the distance between the paired reads is large enough, repeats will be spanned by pairs of reads removing ambiguity from the assembly. The ability to produce increased quantities of paired read sequence data, combined with custom bioinformatics tools, provide the foundation to sequence large and complex plant genomes.

SSAKE was one of the first short-read assemblers released [6]. SSAKE does not incorporate data from paired reads and thus has no way to identify and differentiate between repetitive regions. Indeed in non-strict mode SSAKE will effectively collapse multiple repetitive regions into one contig, and in strict mode any reads at the boundary of a repetitive region will break the contig. Errors in the data set will also break contigs. VCAKE was released in 2007 and is an extension of SSAKE that can handle erroneous reads [7]. In tests between VCAKE and SSAKE, SSAKE and VCAKE perform equally well on simple viral sequences but VCAKE appears to perform much better on more complicated sequences [7]. VCAKE does not make use of paired read data and so also cannot resolve ambiguities caused by repetitive regions. SHARCGS builds on the methods used in SSAKE and VCAKE [8]. The default behaviour for SHARCGS produces three naïve assemblies with different filtering levels. SHARCGS then attempts to merge contigs from different rounds of assembly to produce a more accurate set of contigs which are returned to the user with quality values. As with both SSAKE and VCAKE, SHARCGS does not include paired read information in the algorithm and as a result has an increased probability of collapsing repetitive regions.

Edena was the first short-read assembly algorithm to be released that uses the traditional overlap-layout-consensus approach [9]. Edena does not include an error correction phase before graph generation, which leads to the formation of a messy sequence graph; however, it does include a three step error correction phase, which cleans the graph before assembly begins. ALLPATHS [10] begins by correcting errors and then makes a large set of assemblies, referred to as unipaths. ALLPATHS then uses paired read information to sort unipaths into localized groups. For each localized group of unipaths, the number of paths is trimmed and edited using cleaned paired read information, removing 'hanging ends' ambiguous loops and collapsed edges until, ideally only one path remains. This method reduces the complexity in the overall graph by making local optimizations. The results from the local optimizations are stitched together to produce one long sequence graph. Unlike every other algorithm described here, it returns the entire graph to the user as opposed to just the contigs. Two further iterations of the ALLPATHS algorithm have since been published; ALLPATHS2 [11] and ALLPATHS-LG [12]. ALLPATHS2 uses essentially the same algorithm as the original ALLPATHS but includes newer methods for handling and classifying *k*-mers. ALLPATHS-LG is the latest iteration of the algorithm. This version incorporates improvements in its use of long insert size data and error correction methods which increase its ability to process and assemble low coverage and repetitive regions in the target.

Velvet is the name given to a collection of algorithms for assembling NGS data, first released in 2008 [13]. Since its initial release, both Zerbino and the community at large have made many contributions and improvements to the original code base. Velvet uses an Eulerian model where *k*-mers are stored as edges in a partial de Bruijn graph. Like Edena, Velvet does not include an initial error correction phase but instead uses a series of error correction algorithms to clean up the resulting graph. These algorithms work in a method analogous to the error correction phase in Edena where tips are removed and then bubbles. In Velvet, tips are removed only if they are shorter than $2k$, where k is the *k*-mer length. However, the use of a de Bruijn graph, while highly efficient in terms of memory use, appears to further complicate the P-bubble removal step.

The original Velvet algorithm included a step called Tour Bus which traverses the graph looking for P-bubbles, and when they are found uses a combination of coverage and topographical information to remove the erroneous edges. Velvet then assumes that all remaining low copy number edges must be

errors and removes them from the graph. Finally, the algorithm uses paired read information to join and merge naïve contigs and ultimately produce a collection of scaffolds which are returned to the user. Updates to the original algorithm [14] include improved scaffold formation and error correction as well as memory and speed improvements. In 2009, Zerbino updated the algorithm to include two new algorithmic steps: Rock Band, which improves resolution of repetitive regions; and Pebble, which improves Velvet's ability to make use of reads of different lengths. Velvet was originally developed for relatively small genomes but has been successfully applied for large and complex genomes. Larger genomes require substantial amounts of memory, sometimes in excess of 1 TB RAM, and several modifications of the software have been applied specifically to improve memory efficiency. The latest update of Velvet introduced the parallelization of a number of steps in the algorithm which reduced its total running time. This update was developed by members of the sequencing community and is as yet unpublished. Velvet is currently the most popular assembler available and the original article has been cited 348 times (ISI index, as of 4 July 2011), more than any other competing assembler and has been applied to sequencing projects such as bread wheat [23].

ABySS (Assembly By Short Sequences) is an assembly algorithm released in 2009, which works in very similar way to Velvet. The main goal of ABySS is to address scalability limitations (memory and time), which were common in the then available array of assembly algorithms [15]. The primary innovation of ABySS is the ability to store a de Bruijn graph in a distributed fashion, allowing assembly to proceed in parallel across a networked collection of computers. Unlike Velvet, which requires substantial memory on a single large node, ABySS reduces the memory requirement by distribution across multiple smaller nodes, a more usual form of computer infrastructure. The ABySS algorithm was shown to successfully handle large data sets while its accuracy was shown to be very similar to that produced by Velvet. Recent improvements to Velvet have addressed many of the scalability issues which ABySS was developed to address.

The latest addition to the EULER family of algorithms [16] is EULER SR [17] which has been optimized to assemble short reads. The original EULER algorithm was designed as an implementation of the Idury and Waterman algorithm but included a novel method for error correction. After the reads have been corrected, a de Bruijn graph is constructed and a set of contigs is produced by assembly and returned to the user. Like Velvet, Euler can make use of paired read data to assist in the assembly.

A recent addition to the collection of *de novo* assemblers is SOAPdenovo, released in 2010 as part of the SOAP (Short Oligonucleotide Analysis Package) collection of alignment and assembly programs developed at the BGI [18, 19]. SOAPdenovo has been successfully applied to a number of high profile genome sequencing projects including the Giant Panda genome sequencing project [20] and the sequencing of the genome of an Asian individual [21] to crops such as *Brassica rapa* [22]. Like many other short-read assemblers, SOAPdenovo makes use of a de Bruijn graph and the algorithm is typical of a de Bruijn assembler. Unlike Velvet, SOAPdenovo includes an error correction phase before assembly begins. It then builds a de Bruijn graph and removes inconsistencies including dead end paths (called tips), low-coverage regions, tiny repeats and bubbles. Unfortunately, the article that describes SOAPdenovo does not provide much detail on how these processes are performed. The remainder of the steps are very similar to those employed in Velvet. A set of naïve contigs are produced and scaffolded. Finally the algorithm uses paired read information to fill in the gaps which lie between contigs in the scaffolds.

SaSSY (Short-read ASSemblY) is the only short-read assembler to incorporate paired read information directly during graph formation. The SaSSY algorithm is split into four stages: data loading and overlap calculation, contig building, graph extension and finishing. SaSSY is an overlap-layout assembler based on the method first introduced in [24]; however, it uses novel heuristics to avoid performing all versus all comparison during overlap calculation in the first stage. In the second stage, contigs are formed by identifying and walking along linearly connected subgraphs in the overlap graph. These contigs are then used as seed sequences in later stages. The third stage uses a novel method to extend or grow the contigs. The growing process incorporates both overlap and paired read information to produce a more complex form of the traditional overlap graph called an overlap-pair graph. Nodes in the overlap-pair graph represent paired reads in the

original data. The overlap-pair graph is very similar to 'traditional' overlap graphs and graph cleaning methods designed for those graphs are used in SaSSY with one major exception; SaSSY does not use coverage information to de-convolute repetitive regions. When a graph forks at the boundary of a repetitive region, SaSSY chooses the optimum branch to retain by calculating the distributions of insert sizes and paired read orientations for each candidate branch. In the final stage SaSSY uses paired read information to link together sections of the assembly which are broken at regions of very low or no coverage. These scaffolds are returned to the user. SaSSY has been developed for relatively small but complex sequences such as BACs from repetitive plant genomes and memory requirements increase substantially with larger genome sizes, though further optimization of the code may improve memory usage.

PE Assembler (Paired End Assembler) is one of the latest short-read *de novo* assemblers to be released [25]. The assembler uses a method which is somewhat similar to the SaSSY algorithm described above but with a number of key differences. Similar to SaSSY and unlike most of the other short-read assemblers described here, PE Assembler is an overlap-layout assembler and does not use de Bruijn graphs. The algorithm consists of the following steps: read screening, seed building, contig extension, scaffolding and gap filling. PE Assembler does not include a preliminary error correction step, but uses a read screening process to create a collection of trusted reads which are used to create the initial seeds. The method used is very similar to the error correction algorithm originally proposed by Pevzner [16]. Note that unlike error correction filtering methods which permanently remove reads from the data set, the screening process puts these reads aside temporarily so that the algorithm can make use of them at a later stage. Seeds are constructed using 3′-overlapping information, and irregularities are identified and removed using paired read information. This step is novel to PE Assembler. The contig extension phase is conceptually very similar to the method employed by SaSSY however, unlike SaSSY, PE Assembler does not try to extend seed contigs through repetitive regions. Seed contigs are extended using the basic 3′-extension principle using paired reads where one end overhangs the current seed and the other end lies within the seed. There are a number of cases where there are multiple feasible candidate reads which could be used for extension. These situations can be caused either by errors or by repetitive regions in the target. PE Assembler has methods that can handle the case where multiple candidates exist due to sequencing errors but will terminate extension in the case that the cause is due to repetitive regions. The scaffolding and finishing stages are very similar to those used in many other short-read assembly algorithms.

MAPPING READS TO A REFERENCE

The sheer volume of NGS data generated per run of experiment means that traditional alignment tools relying purely on dynamic programming methods such as the Smith–Waterman algorithm or the BLAST family of alignment heuristics are unable to cope. To meet the requirement of efficient and accurate alignment of millions to billions of short-read sequences to a reference, a whole new range of aligners were developed. The general principle of these tools, including BLAT, MAQ, Bowtie, SOAPaligner/SOAP2, BWA and BFAST, is to subdivide the alignment problem into two steps: the first is a heuristic search of candidate alignment locations (hits) by indexing either the read sequences or the reference genome, followed next by performing the actual alignment.

The BLAST-like alignment tool BLAT is an alignment tool that is designed for the ultra-rapid alignment of sequences to a reference genome [26]. Unlike BLAST, which is designed to be heavily configurable for a number of different applications, the applications for BLAT are quite specific. The most common usage of BLAT is for the rapid lookup of a location in a genome by sequence similarity: using BLAT one can find homologous regions in a human-sized genome in a matter of seconds (http://genome.ucsc.edu/FAQ/FAQblat.html). BLAT is used for rapid sequence-based searching in a number of public genome browsers and genome repositories, including The UCSC Genome Browser Database and EnsEMBL [27, 28]. When used for nucleotide searches on a target genome, BLAT loads the genome into memory in an indexed format. This greatly reduces redundancy and allows a representation for an entire human genome to occupy <1 GB of memory. BLAT is tuned to find highly similar sequences (95% or greater) of 40 bp or longer. For sequences from distant species or very

short sequences, either BLAST or some of the newer short-read mapping tools may prove to be more appropriate.

MAQ [29] was designed for mapping tens of millions of short reads to a reference genome efficiently and dealing with the major issue of alignment accuracy. Due to a combination of factors including the short read lengths produced by NGS; repetitive regions in the genome and sequencing errors, some reads will map equally well to multiple positions. MAQ handles this ambiguity by introducing the concept of mapping quality, which is a Phred-like measure of alignment confidence. The algorithm only performs ungapped alignment on single reads but is capable of making use of read pair information and Phred base quality of the read to calculate the error probability of a read aligning to a position, using a Bayesian posterior probability model. The alignment position with the lowest estimated error is reported, and if a read has equally low error probability at multiple positions, MAQ randomly reports one but sets the mapping quality to zero.

Based on the procedures of the BLAT aligner, BFAST (BLAT-like Fast Accurate Search Tool) is another alignment tool which uses reference genome indexing to improve alignment speed [30]. The interesting aspect of BFAST is its pursuit of alignment sensitivity and accuracy at the expense of alignment speed, making it slower than other popular aligners such as Bowtie, BWA and MAQ but faster than BLAT and with lower reported false-mapping rates overall. BFAST's speed gain over BLAT was achieved through improved usage of the reference index and the inclusion of multiple indices in the initial identification of candidate hits. Following the initial search, BFAST favours a Smith–Waterman alignment approach of the read sequences to the candidate locations. This full gapped local alignment allows BFAST to handle SNPs, insertion and/or deletions better than the other aligners.

Bowtie is an alignment programme implementing Burrows–Wheeler indexing for the rapid alignment of short nucleotide sequences to genomic sequence [31]. Bowtie can align short reads to the human genome at a rate of 25 million short reads per CPU hour while using <2 GB of RAM [31]. Notably, the Bowtie algorithm can permit sequence mismatches and is able to be run on multiple CPU cores simultaneously. Advancements since the publication of Bowtie include the support for paired reads, as well as reads represented in colourspace. An advantage of Bowtie is that it has reciprocal support for a variety of other short-read applications, including Samtools [32]. It forms the basis of the RNA-seq alignment tool, Tophat [33], and the related expression analysis software package, Cufflinks [34].

The SOAP2 alignment tool is a redesign of the original SOAP: Short Oligonucleotide Analysis Package application [35, 36]. SOAP2 implements Burrows–Wheeler indexing to dramatically improve performance and memory usage over its predecessor. SOAP2 allows for up to two mismatches in its sequence alignment and has support for paired-read data. In a comparison with Bowtie, SOAP2 has been shown to align reads faster than Bowtie, sacrificing some memory efficiency for speed and giving slightly less comprehensive results [37]. Yet another Burrows–Wheeler based short read alignment tool, BWA is unique for being the first Burrows–Wheeler based alignment tool to support gapped alignment of short reads [38]. It is also able to support both sequence mismatches and reads presented in colourspace. In the comparisons presented by Li *et al.* [38], BWA is shown to perform similarly to SOAP2 and Bowtie, although performing gapped alignments can somewhat slow down the process. The support for gapped alignment becomes of greater relevance as sequencing read lengths increase and are more likely to contain indels when aligned to divergent genomes and sequences.

The Short-Read Mapping Package (SHRiMP) is a set of algorithms and methods to map short reads against a target genome, even in the presence of a large amount of polymorphism. The main data input of this package is Applied Biosystem's colourspace. For the initial stages, the SHRiMP algorithm searches for *k*-mer matches and performs vectorized Smith–Waterman strictly in colour-space. The algorithm then uses a spaced seed that allows for two adjacent mismatching colours between the read and the reference genome to identify SNPs in colourspace data [39]. This package has been updated to SHRiMP2 which supports more formats, such as Fasta and Fastq input, SAM output, Illumina/Solexa, Roche/454 and AB/SOLiD reads, and supports paired read mapping, miRNA mapping and parallel computation [40].

The inverse of mapping large numbers of reads to a reference, TAGdb is a web-based query tool for aligning query sequences to an existing database of

paired short-read data [41]. The system has been developed using Perl and MySQL and runs on a public web server (http://flora.acpfg.com.au/tagdb/). The interface allows researchers to upload or input a FASTA formatted nucleotide sequence up to 5000 bp long for comparison with one or more paired read sequence libraries. The input sequence is aligned with short reads of significant identity using MEGABLAST [42], and the results visualized using a custom web interface. Each submitted job has a unique identifier, and an email is sent to the user once the job has completed. The processing time per search varies depending on the length of the input sequence and number of matching reads, but generally, searches are completed and results are returned within 20 s to 5 min. TAGdb is currently the only web-based tool for searching short paired read sequence data and it currently hosts data for a variety of plant and fungal species. Additional Illumina paired short-read sequence data may be hosted on request.

MOLECULAR GENETIC MARKER DISCOVERY

Molecular marker technology has developed rapidly over the last decade and two forms of sequence based marker, simple sequence repeats (SSRs), and single nucleotide polymorphisms (SNPs) now predominate applications in modern genetic analysis. Next-generation sequencing has led to the availability of data that enable the mining of SSRs and SNPs, which may then be applied to diversity analysis, genetic trait mapping, association studies and marker-assisted selection [43].

The challenge of *in silico* SNP discovery is not the identification of polymorphic nucleotide positions, but the differentiation of true inter varietal polymorphisms from the abundant sequence errors. This is particularly true for next-generation sequence data which generally has a higher error rate than traditional DNA sequencing. Next-generation sequencing remains prone to inaccuracies as frequent as one error every 20 bp, though this varies substantially between technologies, platforms as well as over time, with companies continuously improving accuracy while increasing read lengths. Read errors impede the electronic mining of this sequence to identify potentially biologically relevant polymorphisms. There are several different types of error that need to be taken into account when differentiating between sequence errors and true polymorphisms, and the approach is highly dependent on the next-generation sequencing platform used to generate the data, as they each have their distinct error profiles. A major source of sequence error comes from the fine balance between the desire to obtain the greatest sequence length, and the confidence that bases are called correctly. Because of this, sequence trimming, filtering and further processing is often applied to reduce the abundance of erroneous sequences. A second cause of error which is particularly an issue with the short reads produced by next-generation sequencing technology is the incorrect mapping of sequences to a reference. This can occur at any genomic region which has two or more similar copies in the genome, due to the presence of multigene families, genome duplications or polyploidy. Software developed for SNP identification from Sanger sequence data is usually not suitable for next-generation sequence data due to the volume of data, incorrect estimation of homopolymer length and problems with base calling quality scores (Pop and Salzburg, 2008).

Pyrobayes, [44] is a modification of the SNP discovery software PolyBayes [45], designed for pyrosequencing reads from 454 sequencing technology. Pyrobayes permits accurate SNP calling in re-sequencing applications, even in shallow read coverage, primarily because it produces more confident base calls than the native base calling program. The method has been demonstrated in *Drosophila* and *Escherichia coli*, but can be applied to any organism for which suitable 454 data are available. Roche also produce two software applications which can be used for SNP identification from 454 data. The GS Reference Mapper Software generates a consensus DNA sequence through mapping and alignment of sequence reads to a reference sequence. The program also generates a list of high-confidence SNPs by identification of individual bases that differ between the generated consensus DNA sequence and the reference sequence. This application can be used for re-sequencing applications in any organism where a reference genome is available for alignment and has been applied for the sequencing of *Eucalyptus grandis* [46]. For discovery of rare SNPs, the GS Amplicon Variant Analyser Software can be used to compare amplicon sequencing reads to a reference sequence. Through very deep amplicon sequencing, hundreds of clonal amplicon reads can be compared to a reference sequence. This permits the detection of

rare sequence variations even in heterogeneous samples, allowing SNP discovery at a population level.

In one of the first examples of cereal SNP discovery from next-generation genome sequence data, Barbazuk *et al.* identified more than 7000 candidate SNPs between maize lines B73 and Mo17, with over 85% validation rate [47]. This success is particularly impressive considering the complexity of the maize genome and the early version of Roche 454 sequencing applied which produced an average read length of only 101 bp. More recent work has attempted to reduce SNP miscalling due to sequence errors [48] and has the potential to improve cereal SNP prediction accuracy using Roche 454 data.

The SNP discovery software AutoSNP [49, 50] has been extended to produce the recently developed AutoSNPdb [51, 52], combining the SNP discovery pipeline of autoSNP with a custom relational database hosting information on the polymorphisms, cultivars and gene annotations. This enables efficient mining and interrogation of the data, and users may search for SNPs within genes with specific annotation or for SNPs between defined cultivars. AutoSNPdb can integrate both Sanger and Roche 454 pyrosequencing data enabling efficient SNP discovery from next-generation sequencing technologies. AutoSNPdb principally uses redundancy to differentiate between sequence errors and real SNPs. A co-segregation score is also calculated based on whether multiple SNPs define a haplotype, and this is used as a second, independent measure of confidence.

The larger data volumes from the Illumina-sequencing platform enable the confident discovery of very large numbers of genome-wide SNPs [53]. More than 1 million SNPs have been identified between six inbred maize lines [54]. This study also identified a large number of presence/absence variations (PAVs), which may be associated with heterosis in this species. SNPs are naturally more prevalent when diverse germplasm is used for their discovery. Around 3.6 million SNPs were identified by sequencing 517 rice landraces [55]. This detailed study allowed the association of genome variation with complex traits in rice and is a model for future studies in the larger genome cereals. Analysis of genome-wide SNPs in soybean identified large linkage blocks which would be assayed using a limited number of SNPs [54]. The characterization of linkage blocks assists the introduction of genomic selection methods for cereal crop improvement [56].

The discovery of SSRs has changed rapidly over the years. Initially, the discovery of SSR loci required the construction of genomic DNA libraries enriched for SSR sequences, followed by DNA sequencing [57]. Subsequently, the increasing availability of sequence data made it more economical and efficient to use computational tools to identify SSR loci. Several computational tools were developed for the identification of SSRs within sequence data as well as for the design of PCR amplification primers [58, 59]. The identification of SSRs from next-generation sequence data has proven to be more challenging than the identification of SNPs, primarily due to the short length of the reads. However there have been some successes. SSRs for various species were identified from Roche 454 GS FLX data using two SSR enrichment methods [60]. Using only 16th of a Roche 454 GS FLX plate, Abdelkrim *et al.* produced 17 215 short sequence reads with an average size of 243 bp [61]. Bioinformatics analysis identified 231 reads containing SSRs, of which only 24 were suitable for PCR primer design. This highlights the challenge of discovering SSR markers from short reads, though SSRs have been identified from Illumina reads as short as 80 bp [62]. The increased read length of some of the third-generation sequencing technologies is likely to greatly increase the utility of this data for SSR marker discovery.

NEXT-GENERATION TRANSCRIPTOMICS

TopHat [63] is an open-source software designed to align reads from RNA-Seq to a reference genome without relying on known splice sites. TopHat takes a reference genome (as a Bowtie index) and RNA-Seq reads as FASTA or FASTQ and produces alignments in SAM format. With default parameter values TopHat detects junctions even in genes transcribed at very low levels. TopHat version 1.0.7 and later has been extended to exploit the longer paired reads and align reads across splice junctions. It splits a read 75 bp or longer in three or more segments of approximately equal size (25 bp) and maps them independently. It uses Bowtie (http://bowtie-bio.sourceforge.net) a short-read mapping program [64], to map non-junction reads (those contained within exons) against the reference genome. All reads that do not map to the genome are set aside as 'initially unmapped reads', or IUM reads. Tophat searches the IUM reads in order to find reads that

span junctions for each splice junction. The MAQ [29] assembly module is used to build a consensus of the mapped regions. TopHat is implemented in C^{++} and runs on Linux and Mac OS X. It uses the SeqAn library [65] and requires Bowtie and MAQ.

The Cufflinks assembler [66] is an open-source C^{++} program and runs on Linux and Mac OS X. It can identify complete novel transcripts and probabilistically assign reads to isoforms. Additionally it contains Cuffcompare and Cuffdiff tools. Cuffcompare validates Cufflinks output, transfrags (assembled transcript fragments), against annotated transcriptomes and also finds transfrags common to multiple assemblies. Cuffdiff then performs differential expression testing. This assembler was designed to investigate transcriptional, splicing regulation and find the minimal number of transcripts that 'explain' the reads (that is, every read should be contained in some transcript). Cufflinks takes as input cDNA fragment sequences that have been aligned to the genome e.g. by TopHat, which can align reads across splice junctions without relying on gene annotation in order to produce spliced alignments.

VISUALIZATION TOOLS

Bambino is a viewer for next-generation sequence files, with a focus on variant visualization and detection [67]. The viewer is heavily influenced by consed [68] and presents second-generation sequence data in a very similar style (Figure 1). Unlike consed however, Bambino is a Java tool that is fully cross-compatible with a Java Webstart launcher that allows users to launch the software directly from their website on almost any Java-compatible system. Bambino allows for multiple BAM formatted input data files, and allows users to pool data within the programme, permitting the analysis of variation between samples. Bambino also features a graphical overview display for the given assembly and enables the annotation of UCSC genome annotations on the given contig.

Tablet is another viewer designed for the visualization of next-generation sequence data [69]. Tablet is able to load assembly data from a variety of assembly file formats including ACE, AFG, MAQ, SOAP, SAM and BAM. The latter four require the consensus or reference sequence to be imported separately, usually as a FASTA or FASTQ file. Tablet provides a contig overview as well as a sequence level viewer, has the ability to highlight disagreements from the reference or consensus sequence in the mapped reads and can highlight reads by a number of criteria (Figure 2). These criteria allow for read colouring by nucleotide, read type or read orientation. A useful feature of Tablet

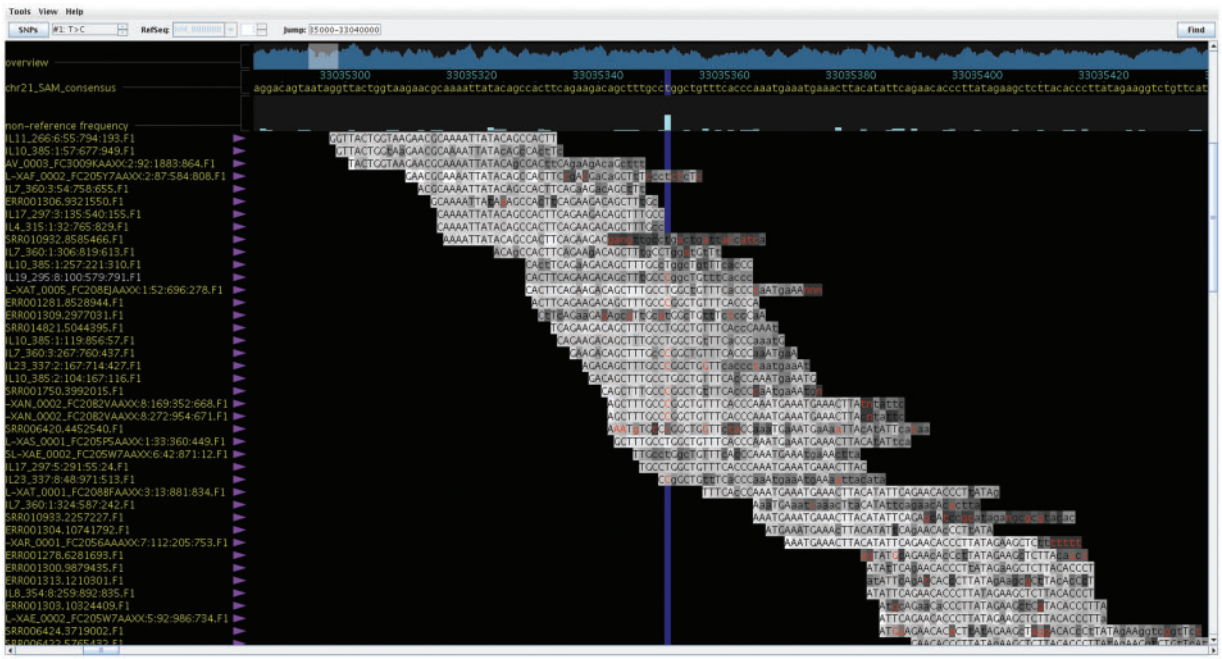

Figure 1: The bambino browser showing a BAM file for reads mapped against human chromosome 21. A SNP between the read set and the consensus sequence is shown in this view as a vertical line. This SNP was defined using the internal Bambino variant caller.

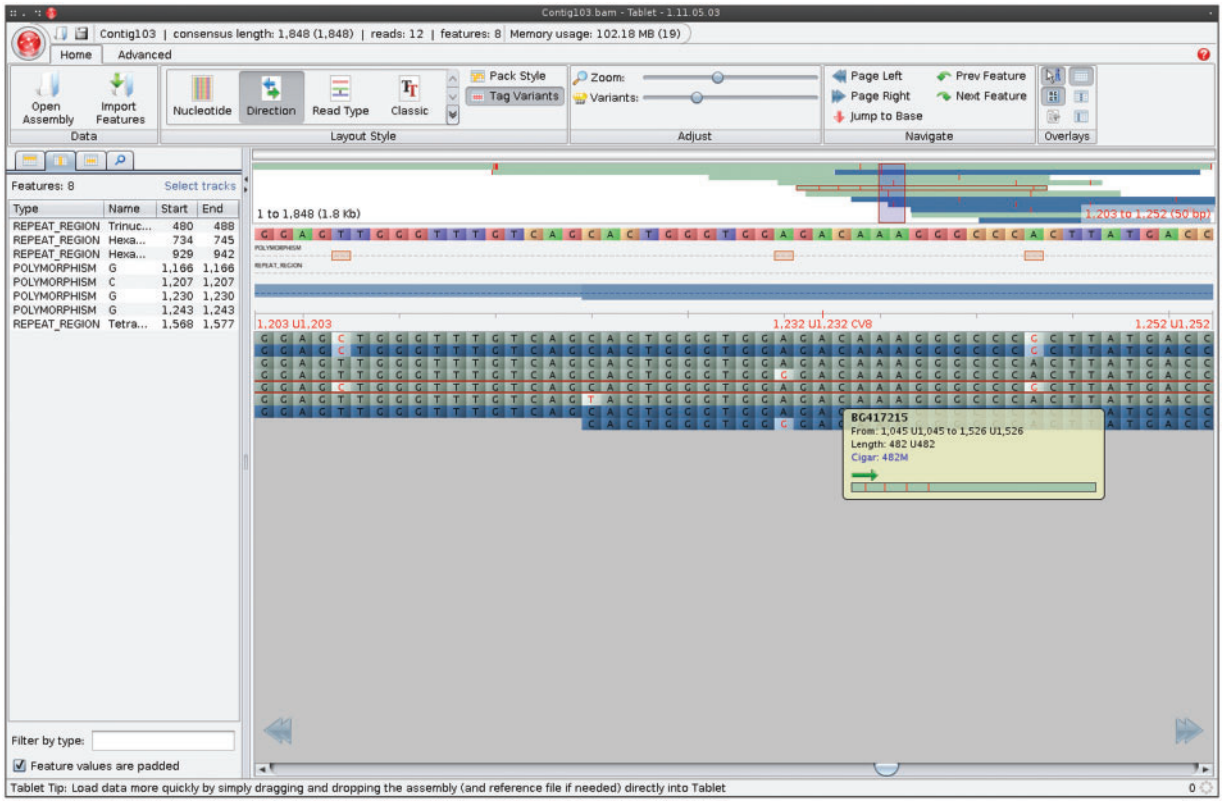

Figure 2: Tablet screenshot visualizing a barley EST assembly. The assembly has been annotated with a GFF feature file of pre-calculated sequence-based markers which are shown as a track above the sequence view (bottom right) and in the tabulated summary space (left).

is the ability to import feature annotations in the form of GFF files, and it also provides summary information for the annotation in addition to information for the contig. Tablet is Java-based and is available for Linux, OSX, Windows and Solaris platforms, in both 32- and 64-bit versions. It supports multi-core processor architectures and allows faster navigation of NGS data with low memory usage. Tablet has been implemented as a hybrid system that provides the advantages of memory-based (where all the data are loaded into memory) and disk-cached (only the visible segment of the data set is loaded in the memory, and the remainder are stored on the disk) methods.

The Integrative Genomics Viewer (IGV) is an open-source visualization tool (http://www.broadinstitute.org/igv/) that facilitates exploration of diverse, large-scale genomic data sets [70]. The architecture of IGV enables fast data loading including NGS data while consuming minimal resources and allows the user to zoom across the genome at any level of detail from whole genome to base pairs including putative SNPs. IGV supports many genomic data types, including array-based data, such as expression and copy-number arrays, RNA interference screens, gene expression, methylation and genomic annotations. IGV can load data from both local as well as remote sources and also allows collaborators to load and share data locally or over the Internet. It is written in Java and is platform independent, with both 32- and 64-bit versions.

The Savant (Sequence Annotation, Visualization and ANalysis Tool) Genome Browser is an open-source desktop visualization and analysis browser for genomic data [71]. It was developed for visualizing and analysing high-throughput sequencing (HTS) data e.g. NGS, with low memory requirements. Users can navigate the data set using keyboard or mouse to zoom, pan and seek using range controls. It can be used to visualize any genome-based sequence; point, interval and continuous data sets, and has multiple visualization modes that enable easy identification of genomic variants (including SNPs, structural and copy number variants), and functional genomic information (e.g. peaks in ChIP-seq data) in the context

of genomic annotations. As input, Savant accepts FASTA, BED, SAM/BAM, WIG, GFF and any tab-delimited text file containing positional annotations. In order to allow fast random access to text files, Savant formats and saves each file to an indexed binary data structure specific to each data type to provide very efficient search operations. Users can extend the application by adding plug-ins for specific tasks e.g. for computing genome-wide statistics, such as the fraction of SNPs in exons. Once developed, plug-ins can be shared among users through the plug-in section of the Savant web site. This tool is written in Java so is platform independent, with both 32- and 64-bit versions.

MagicViewer was developed for short-read alignment visualization and annotation [72]. It requires a reference genome sequence in fasta format, a sorted BAM file containing the aligned short reads and an optional reference genome annotation file in GFF format. MagicViewer allows zoom from whole chromosomes to individual bases. When the mouse hovers on a specific read, a pop-up appears with read ID, location, base quality, read length and orientation. The Genome Analysis Toolkit (GATK) is an open-source software framework to develop analysis tools for next-generation sequencing data [73]. It was built in MagicViewer to identify genetic variation between short reads and reference genomes. MagicViewer allows users to change parameters for heterozygosity, confidence threshold and max coverage. For candidate SNPs, it provides options (thresholds for coverage, quality, variant frequency and number of reads) for display and filtering to remove low confidence SNPs. The output of genetic variation calling is saved in a variable call format [74]. MagicViewer allows users to adjust parameters (primer length, Tm, GC content, product Tm and the number of primers) for Primer3 in order to amplify a specific genomic region.

Geneious [75] is a powerful sequence visualization and analysis tool that allows users to visualize and manipulate next-generation sequence data by performing a number of operations (Figure 3). Geneious also provides tools for the assembly, alignment and annotation of genomic reads and sequence, as well as exploratory alignment against public repositories using the BLAST sequence search capabilities. An interesting visualization feature in

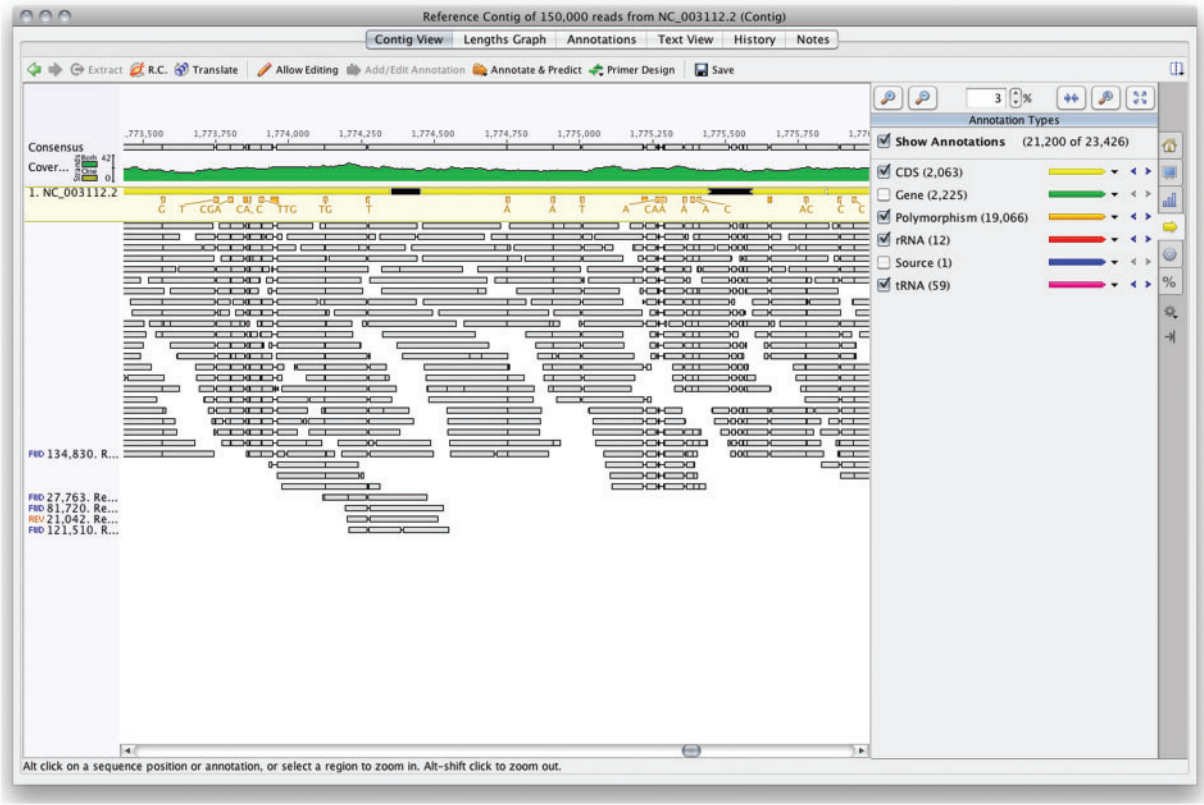

Figure 3: Geneious screenshot visualizing assembled sequence reads, sequence variation and annotation.

Geneious is the ability to create a split view, providing a number of visualization aspects for the same data. Geneious can also be used effectively as a sequence format conversion tool. There are a large number of bioinformatics formats for next-gen sequence data, and an even larger number of tools for loading them. However, many of these tools are either command line based or limited to particular platforms, making conversion between the differing formats difficult and unwieldy. Using a common tool for the conversion of data can help to simplify otherwise complicated workflows. Geneious is a cross-platform tool written in Java, with packages available for Linux, OSX, Windows and Solaris.

DATA STORAGE

The recent revolution in DNA-sequencing technology that has brought down the cost of DNA sequencing has led to an explosion on sequence data volumes being generated. There has also been a dramatic shift in the type of sequence data being generated, with vast numbers of short reads or pairs of short reads replacing the traditional relatively long reads produced by Sanger sequencing. These changes in data quantity and format has led to a rethinking of sequence data management, storage and visualization and provided a major challenge for bioinformatics. The vast amount of sequence data that is currently being generated cannot be stored in the same manner as traditional sequence data and innovative approaches are required to manage the change in what is stored and how users query the information [76]. As genome sequencing technology continues to advance, it is unlikely that all the raw data will continue to be stored, but rather new databases which provide interfaces to processed data such as sequence variants or functional annotations will be developed, along with species or specific purpose (e.g. disease related) databases.

CONCLUSIONS

The growth of next-generation sequencing technology has been matched by a parallel growth in associated bioinformatics tools. These tools continue to increase in number and complexity. It is important that users are aware of the differences between apparently similar tools if they are to make informed decisions and undertake the most appropriate analysis.

Key Points

- NGS is revolutionizing the way genomics studies are approached and opens new avenues to understand how the genome influences the observed phenotype.
- Many bioinformatics tools have been developed to analyse the massive amount of data generated by NGS applications. These tools predominantly aim to accelerate data analysis and assist interpretation.
- New software and algorithms are being developed continuously, with improving speed, efficiency and the ability to handle the specific characteristics of NGS data.

Acknowledgements

The authors would like to acknowledge the funding support. C.D. is a PhD student under the supervision of D.E. and is currently employed as a bioinformatics software developer by Biomatters Ltd.

FUNDING

The Australian Research Council (Projects: LP0882095, LP0883462 and DP0985953); the Australian Genome Research Facility (AGRF); the Queensland Cyber Infrastructure Foundation (QCIF); the Australian Partnership for Advanced Computing (APAC) and Queensland Facility for Advanced Bioinformatics (QFAB). D.E. receives research funds from Biomatters Ltd., producers of Geneious software.

References

1. Edwards D, Batley J. Plant genome sequencing: applications for crop improvement. *Plant Biotechnol J* 2010;**7**:1–8.
2. Imelfort M, Edwards D. De novo sequencing of plant genomes using second-generation technologies. *Brief Bioinform* 2009;**10**:609–18.
3. Imelfort M, Batley J, Grimmond S, *et al*. Genome sequencing approaches and successes. In: Somers D, Langridge P, Gustafson J (eds). *Plant Genomics*. USA: Humana Press, 2009:345–58.
4. Lin Y, Li J, Shen H, *et al*. Comparative studies of de novo assembly tools for next-generation sequencing technologies. *Bioinformatics* 2011;**27**:2031–7.
5. Zhang W, Chen J, Yang Y, *et al*. A practical comparison of de novo genome assembly software tools for next-generation sequencing technologies. *Plos One* 2011;**6**: e17915.
6. Warren RL, Sutton GG, Jones SJM, *et al*. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics* 2007;**23**:500–1.

7. Jeck WR, Reinhardt JA, Baltrus DA, *et al*. Extending assembly of short DNA sequences to handle error. *Bioinformatics* 2007;**23**:2942–4.
8. 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**:1697–706.
9. 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**:802–9.
10. Butler J, MacCallum I, Kleber M, *et al*. ALLPATHS: de novo assembly of whole-genome shotgun microreads. *Genome Res* 2008;**18**:810–20.
11. MacCallum I, Przybylski D, Gnerre S, *et al*. ALLPATHS 2: small genomes assembled accurately and with high continuity from short paired reads. *Genome Biol* 2009;**10**: R103.
12. Gnerre S, MacCallum I, Przybylski D, *et al*. High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc Natl Acad Sci USA* 2011;**108**: 1513–8.
13. Zerbino DR, Birney E. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res* 2008;**18**:821–9.
14. Zerbino DR, McEwen GK, Margulies EH, *et al*. Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. *Plos One* 2009;**4**(12):e8407.
15. Simpson JT, Wong K, Jackman SD, *et al*. ABySS: a parallel assembler for short read sequence data. *Genome Res* 2009;**19**: 1117–23.
16. Pevzner PA, Tang HX, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA* 2001;**98**:9748–53.
17. Chaisson MJ, Pevzner PA. Short read fragment assembly of bacterial genomes. *Genome Res* 2008;**18**:324–30.
18. Li RQ, Li YR, Kristiansen K, *et al*. SOAP: short oligonucleotide alignment program. *Bioinformatics* 2008;**24**: 713–4.
19. Li RQ, Zhu HM, Ruan J, *et al*. De novo assembly of human genomes with massively parallel short read sequencing. *Genome Res* 2010;**20**:265–72.
20. Li RQ, Fan W, Tian G, *et al*. The sequence and de novo assembly of the giant panda genome. *Nature* 2010; **463**:1106.
21. Wang J, Wang W, Li R, *et al*. The diploid genome sequence of an Asian individual. *Nature* 2008;**456**:60–5.
22. The Brassica rapa Genome Sequencing Project Consortium. The genome of the mesopolyploid crop species Brassica rapa. *Nature Genet* 2011;**43**:1035–40.
23. Berkman BJ, Skarshewski A, Lorenc MT, *et al*. Sequencing and assembly of low copy and genic regions of isolated Triticum aestivum chromosome arm 7DS. *Plant Biotechnol J* 2011;**9**:768–75.
24. Myers EW. Toward simplifying and accurately formulating fragment assembly. *J Comput Biol* 1995;**2**:275–90.
25. Ariyaratne PN, Sung W-K. PE-Assembler: de novo assembler using short paired-end reads. *Bioinformatics* 2011; **27**:167–74.
26. Kent W. BLAT - the BLAST-like alignment tool. *Genome Res* 2002;**12**:656–64.
27. Karolchik D, Kuhn RM, Baertsch R, *et al*. The UCSC Genome Browser Database: 2008 update. *Nucleic Acids Res* 2008;**36**:D773–9.
28. Flicek P, Amode MR, Barrell D, *et al*. Ensembl 2011. *Nucleic Acids Res* 2011;**39**:D800–6.
29. Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res* 2008;**18**:1851–8.
30. Homer N, Merriman B, Nelson SF. BFAST: an alignment tool for large scale genome resequencing. *Plos One* 2009;**4**: e7767.
31. Langmead B, Trapnell C, Pop M, *et al*. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* 2009;**10**:R25.
32. Li H, Handsaker B, Wysoker A, *et al*. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* 2009; **25**:2078–9.
33. Trapnell C, Pachter L, Salzberg SL. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* 2009;**25**: 1105–11.
34. Trapnell C, Williams BA, Pertea G, *et al*. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotechnol* 2010;**28**:511–5.
35. Li R, Li Y, Kristiansen K, *et al*. SOAP: short oligonucleotide alignment program. *Bioinformatics* 2008;**24**:713–4.
36. Li R, Yu C, Li Y, *et al*. SOAP2: an improved ultrafast tool for short read alignment. *Bioinformatics* 2009;**25**:1966–7.
37. Horner DS, Pavesi G, Castrignano T, *et al*. Bioinformatics approaches for genomics and post genomics applications of next-generation sequencing. *Brief Bioinformatics* 2010;**11**: 181–97.
38. Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* 2009;**25**: 1754–60.
39. Rumble S, Lacroute P, Dalca A, *et al*. SHRiMP: accurate mapping of short color-space reads. *PLoS Comput Biol* 2009; **5**:1000386e.
40. David M, Dzamba M, Lister D, *et al*. SHRiMP2: sensitive yet practical short read mapping. *Bioinformatics* 2011;**27**: 1011–2.
41. Marshall D, Hayward A, Eales D, *et al*. Targeted identification of genomic regions using TAGdb. *Plant Methods* 2010; **6**:19.
42. Zhang Z, Schwartz S, Wagner L, *et al*. A greedy algorithm for aligning DNA sequences. *J Comput Biol* 2000;**7**: 203–14.
43. Duran C, Appleby N, Edwards D, *et al*. Molecular genetic markers: discovery, applications, data storage and visualisation. *Curr Bioinform* 2009;**4**:16–27.
44. Quinlan AR, Stewart DA, Stromberg MP, *et al*. Pyrobayes: an improved base caller for SNP discovery in pyrosequences. *Nat Methods* 2008;**5**:179–81.
45. Marth GT, Korf I, Yandell MD, *et al*. A general approach to single-nucleotide polymorphism discovery. *Nat Genetics* 1999;**23**:452–6.
46. Novaes E, Drost DR, Farmerie WG, *et al*. High-throughput gene and SNP discovery in Eucalyptus grandis, an uncharacterized genome. *BMC Genomics* 2008;**9**:312.

47. Barbazuk WB, Emrich SJ, Chen HD, *et al*. SNP discovery via 454 transcriptome sequencing. *Plant Journal* 2007;**51**: 910–18.
48. Brockman W, Alvarez P, Young S, *et al*. Quality scores and SNP detection in sequencing-by-synthesis systems. *Genome Res* 2008;**18**:763–70.
49. Barker G, Batley J, O'Sullivan H, *et al*. Redundancy based detection of sequence polymorphisms in expressed sequence tag data using autoSNP. *Bioinformatics* 2003;**19**:421–2.
50. Batley J, Barker G, O'Sullivan H, *et al*. Mining for single nucleotide polymorphisms and insertions/deletions in maize expressed sequence tag data. *Plant Physiol* 2003;**132**:84–91.
51. Duran C, Appleby N, Clark T, *et al*. AutoSNPdb: an annotated single nucleotide polymorphism database for crop plants. *Nucleic Acids Res* 2009;**37**:D951–3.
52. Duran C, Appleby N, Vardy M, *et al*. Single nucleotide polymorphism discovery in barley using autoSNPdb. *Plant Biotechnol J* 2009;**7**:326–33.
53. Imelfort M, Duran C, Batley J, *et al*. Discovering genetic polymorphisms in next-generation sequencing data. *Plant Biotechnol J* 2009;**7**:312–7.
54. Lai JS, Li RQ, Xu X, *et al*. Genome-wide patterns of genetic variation among elite maize inbred lines. *Nat Genet* 2010;**42**: 1027–158.
55. Huang XH, Wei XH, Sang T, *et al*. Genome-wide association studies of 14 agronomic traits in rice landraces. *Nat Genet* 2010;**42**:961–76.
56. Duran C, Eales D, Marshall D, *et al*. Future tools for association mapping in crop plants. *Genome* 2010;**53**: 1017–23.
57. Edwards KJ, Barker JHA, Daly A, *et al*. Microsatellite libraries enriched for several microsatellite sequences in plants. *Biotechniques* 1996;**20**:758.
58. Robinson AJ, Love CG, Batley J, *et al*. Simple sequence repeat marker loci discovery using SSR primer. *Bioinformatics* 2004;**20**:1475–6.
59. Jewell E, Robinson A, Savage D, *et al*. SSRPrimer and SSR Taxonomy Tree: Biome SSR discovery. *Nucleic Acids Res* 2006;**34**:W656–9.
60. Santana QC, Coetzee MPA, Steenkamp ET, *et al*. Microsatellite discovery by deep sequencing of enriched genomic libraries. *Biotechniques* 2009;**46**:217–23.
61. Abdelkrim J, Robertson BC, Staton JL, *et al*. Fast, cost-effective development of species specific microsatellite markers by genomic sequencing. *Biotechniques* 2009;**46**: 185–192.
62. Jennings TN, Knaus BJ, Mullins TD, *et al*. Multiplexed microsatellite recovery using massively parallel sequencing. *Mol Ecol Resour* 2011;**11**(6):1060–67.
63. Trapnell C, Pachter L, Salzberg SL. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* 2009;**25**: 1105–11.
64. Langmead B, Trapnell C, Pop M, *et al*. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* 2009;**10**:R25.
65. Doering A, Weese D, Rausch T, *et al*. SeqAn An efficient, generic C++ library for sequence analysis. *BMC Bioinformatics* 2008;**9**:11.
66. Trapnell C, Williams BA, Pertea G, *et al*. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotech* 2010;**28**:511–5.
67. Edmonson MN, Zhang J, Yan C, *et al*. Bambino: a variant detector and alignment viewer for next-generation sequencing data in the SAM/BAM format. *Bioinformatics* 2011;**27**: 865–6.
68. Gordon D, Abajian C, Phil G. Consed: a graphical tool for sequence finishing. *Genome Res* 1998;**8**:195–202.
69. Milne I, Bayer M, Cardle L, *et al*. Tablet–next generation sequence assembly visualization. *Bioinformatics* 2010;**26**: 401–2.
70. Robinson JT, Thorvaldsdottir H, Winckler W, *et al*. Integrative genomics viewer. *Nat Biotech* 2011;**29**:24–6.
71. Fiume M, Williams V, Brook A, *et al*. Savant: genome browser for high-throughput sequencing data. *Bioinformatics* 2010;**26**:1938–44.
72. Hou H, Zhao F, Zhou L, *et al*. MagicViewer: integrated solution for next-generation sequencing data visualization and genetic variation detection and annotation. *Nucleic Acids Res* 2010;**38**:W732–6.
73. McKenna A, Hanna M, Banks E, *et al*. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res* 2010; **20**:1297–303.
74. Danecek P, Auton A, Abecasis G, *et al*. The variant call format and VCFtools. *Bioinformatics* 2011;**27**: 2156–8.
75. Drummond AJ, Ashton B, Buxton S, *et al*. Geneious v5.4. http://www.geneious.com/.
76. Batley J, Edwards D. Genome sequence data: management, storage, and visualization. *Biotechniques* 2009;**46**: 333–6.