BIOINFORMATICS APPLICATIONS NOTE Vol. 25 no. 18 2009, pages 2434–2435
doi:10.1093/bioinformatics/btp403

Sequence analysis

SNP-o-matic

Heinrich Magnus Manske* and Dominic P. Kwiatkowski
Wellcome Trust Sanger Institute, Hinxton, Cambridge, CB10 1SA, UK

Received on March 27, 2009; revised on June 12, 2009; accepted on June 27, 2009
Advance Access publication July 2, 2009
Associate Editor: Limsoon Wong

ABSTRACT

Motivation: High throughput sequencing technologies generate large amounts of short reads. Mapping these to a reference sequence consumes large amounts of processing time and memory, and read mapping errors can lead to noisy or incorrect alignments. SNP-o-matic is a fast, memory-efficient and stringent read mapping tool offering a variety of analytical output functions, with an emphasis on genotyping.

Availability: http://snpomatic.sourceforge.net

Contact: mm6@sanger.ac.uk

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Analysis of genome variation has been revolutionized by the advent of next-generation sequencing technologies (Bentley *et al.*, 2008; Li *et al.*, 2008b; Shendure and Ji, 2008). The short length of sequence reads, e.g. 50 base pairs, can pose considerable challenges in achieving accurate genome alignment, particularly if the genome sequence is highly polymorphic. Discovery of single nucleotide polymorphisms (SNPs) and other variants depends on the alignment algorithm allowing some mismatches to the reference sequence, but allowing too many mismatches may lead to incorrect alignments. Thus the process of discovering novel variants amounts to a complex statistical problem, particularly if sequencing errors and other sources of noise are taken into account. Various discovery algorithms have been developed and this is an area of much research interest (for example MAQ, Li *et al.*, 2008a; and bowtie, Langmead *et al.*, 2009).

Here we focus on the problem of describing the genotype of an individual using short-read sequencing data. In principle, this can be incorporated into the same algorithms used for discovering novel variants, an approach that appears to work well for the human genome (Bentley *et al.*, 2008). However, there are circumstances in which it may be useful to separate SNP discovery from SNP genotyping. For example, SNP discovery in *Plasmodium falciparum* is particularly complicated due to 80% AT content, many repeat sequences, regions of extreme polymorphism and the multiclonality of natural isolates. Thus different SNP discovery algorithms return widely different results. One way of addressing this problem is to begin by annotating the reference genome with all the putative SNPs generated by different discovery algorithms. Then individual SNPs may be genotyped by performing a stringent alignment of the sequencing reads against the reference genome, allowing for all the putative variable positions.

*To whom correspondence should be addressed.

To support this sort of genotyping analysis, we developed SNP-o-matic as a fast way of mapping short sequence reads to a reference genome with a list of putative variable positions that are specified at the outset. The default settings are highly stringent, returning only those sequence reads that align perfectly with the reference genome after allowing for the putative variable positions.

An important feature of SNP-o-matic, which allows the rapid processing of large volumes of sequencing data, is that the reference genome sequence is first indexed (on the fly or by using a pre-computed index from disk), and then each sequence read or read pair is examined one at a time. This avoids having to build and store an index of the reads saving both compute cycles and memory. Indexing of the reference genome is done in memory on the fly from a generic FASTA file. A list of putative SNPs, if supplied, is integrated into the reference before indexing, and all permutations of this SNP-containing sequence are indexed. Indexing the 25 Mb *P. falciparum* genome (without SNPs) takes about 30 s on a single CPU core and occupies ∼1 GB of memory. A memory-saving option can reduce both memory and indexing time significantly at the expense of a longer mapping phase. The index can be stored in a file for future use, further reducing the time required for this step, or to facilitate the analysis for larger genomes (Supplementary Material).

Reads are supplied in either FASTA or FASTQ (http://maq.sourceforge.net/fastq.shtml) format; read pairs can be in either single or split files. In performance tests, mapping 10-million 37 base paired reads against the *P. falciparum* genome takes 70 s on a single CPU core, not counting the indexing. No additional memory is required for the mapping. Additional time and memory may be required for some of the output functions.

For genotyping, a variable length indexed kmer (default 26 bases) is compared to the same length kmer for each read (or both reads in a read pair). Matches in these bases thus have to be perfect, with respect to the putative SNPs. The remaining bases of the read are then compared base-by-base to the reference. By default, these matches have to be perfect as well, but a limited number of mismatches can be allowed. This stringency will avoid false SNP calls in genotyping mode that would otherwise be caused by aligning reads containing sequencing errors. Thus, SNP-o-matic will generally map less reads than other algorithms, but the mapping will have much higher accuracy. When allowing mismatches, the kmer length can be varied to increase mapping tolerance (Supplementary Material).

Both parts of a read pair have to map on the same chromosome for valid mapping; a fragment range can be used to limit their mapping distance to conform to the expected size distribution for the library. An optional mode can increase stringency by ensuring that at least one read of a read pair maps uniquely within reference the genome.

By default, SNP-o-matic will find and use all valid mappings for a read or read pair within the reference.

When using read pairs, the stringent mapping algorithm can sometimes map one of the reads in the pair, but not the other. SNP-o-matic can output various data about such read pairs. From the mapping position, orientation, and fragment size, a likely position can be estimated for the non-mapping read. Based on this information, reads can be grouped by position and assembled to discover variation. Additionally, the estimated area can be searched for mappings with some mismatches, resulting in potential new SNP calls. This output is the primary method used by SNP-o-matic to discover new SNPs and small-scale variation, both of which require further downstream analysis (Supplementary Material). Scripts for such analysis are under development and will eventually be incorporated into the SNP-o-matic package to augment its core function.

Similarly, both reads of the pair may map to the reference, but not on the same chromosome. This information can be used to detect misassemblies. When using (super)contigs as reference sequence, read pairs can thus be used to link contigs together, determine their order, and estimate the size of the gap between two contigs.

An output type of SNP-o-matic is a read bin, a file containing reads grouped by mapping behavior. Bins are a quick and easy way to filter a read set, for example to remove DNA contamination and noise from non-uniquely mapping reads, or to gather non-mapping reads for further study or assembly. Available bins are single mapping reads (uniquely mapped in the genome), multiple-mapping reads, non-mapping reads, and reads containing IUPAC bases (e.g. 'N'); the later are ignored by SNP-o-matic for mapping.

Mapping/alignment output is supplied for

- pileup,
- coverage (base count per position),
- CIGAR format (http://biowiki.org/CigarFormat),
- gff format (http://biowiki.org/GffFormat),
- SAM format (http://samtools.sourceforge.net/) and
- sqlite database (http://www.sqlite.org/).

For accurate SNP genotyping, it is advantageous to take account of sequence quality scores, especially in regions with low coverage. SNP-o-matic can generate an output file showing each instance where a mapped read covers a putative SNP. Each output line contains the read name, allele position on the reference, reference and observed allele, quality score of the allele base, average and minimum quality of both the entire read as well as the five bases on either side of the allele-calling base, and auxiliary data. This data can be further quality filtered, and used to generate a list of non-reference majority alleles.

Other outputs include observed fragment size distribution, insertion/deletion predictions and inversion detection. These can also be determined by alternative algorithms from the aforementioned mapping/alignment outputs.

SNP-o-matic is written in C++ / C (for performance optimization). Compilation with the Intel icpc compiler has shown significant runtime improvements over g++.

We carried out a number of performance tests which are described in the Supplementary Material and briefly summarized below. The initial tests were based on an artificial dataset consisting of a 1mbp reference genome whose AT content (80%) is similar to the *P. falciparum* genome, and a duplicate genome with randomly introduced SNPs and indels. Solexa read pairs (2×37 bases) with random errors (one in five reads) were generated from the altered genome.

SNP-o-matic correctly genotyped the SNPs when they were given as a putative SNP list. As expected, coverage dropped substantially when a SNP list was not supplied, unless the mapping stringency was reduced.

We have not attempted to conduct a comprehensive comparison of the performance of SNP-o-matic with SNP discovery algorithms such as MAQ as it is designed primarily as a tool to be used after the stage of SNP discovery. However, as an illustration of where SNP-o-matic may be useful, we found that, when analyzing clusters of six SNPs in the simulated dataset, MAQ only called two of the SNPs, whereas SNP-o-matic called all six correctly when they were supplied in a putative SNP list.

The current version of SNP-o-matic does not directly detect indels, but can be adapted to do so by using an optional 'wobble function' to identify read pairs where one read maps perfectly but the other does not, and then using an algorithm such as velvet (Zerbino and Birney, 2008) to assemble the non-mapping reads into a contig which is then mapped to the region covering the deletion site using an algorithm such as blat (Kent, 2002). Using this approach, we found that it was possible to detect a five-base deletion that was introduced into the simulated dataset described above.

Finally, in the Supplementary Material, we provide data on the performance of SNP-o-matic on human chromosomes 1, X, and Y. Based on these findings we estimate that processing an entire human genome using a pre-computed index and the memory saving option, mapping the test reads should take $\sim$20 min and 29 GB of RAM. A similar timeframe, with $<$3 GB RAM usage, would be expected for a chromosome-by-chromosome serial execution; this would require an additional, albeit simple, filtering step to ensure uniqueness across the entire genome.

ACKNOWLEDGEMENTS

The authors thank Chris Newbold for several discussions and suggestions, and Gareth Maslen for help with preparing the manuscript.

Funding: Wellcome Trust, Bill and Melinda Gates Foundation and Medical Research Council.

Conflict of Interest: none declared.

REFERENCES

Bentley,D.R. *et al*. (2008) Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, **456**, 53–59.

Kent,W.J. (2002) BLAT: the BLAST-like alignment tool. *Genome Res*., **12**, 656–664.

Langmead,B. *et al*. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol*., **10**, R25.

Li,H. *et al*. (2008a) Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res*., **18**, 1851–1858.

Li,R. *et al*. (2008b) SOAP: short oligonucleotide alignment program. *Bioinformatics* **24**, 713–714.

Shendure,J. and Ji,H. (2008) Next-generation DNA sequencing. *Nat. Biotechnol*., **26**, 1135–1145.

Zerbino,D.R and Birney,E. (2008) Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res*., **18**, 821–829.