SMALT Manual

July 5, 2011
Version 0.5.5

Abstract

SMALT is a pairwise sequence alignment program for the efficient mapping of DNA sequencing reads onto genomic reference sequences. It uses a combination of short-word hashing and dynamic programming. Most types of sequencing platforms are supported including paired-end sequencing reads.

1 Synopsis

smalt TASK [**OPTIONS**] [*INDEX SEQFIL-A* [*SEQFIL-B*]]

Available tasks

smalt index [**INDEX-OPTIONS**] *INDEX REFSEQ-FILE*
: builds a hash index of k-mer words in the reference sequences and stores it on disk. Two files are written to disk: *INDEX.smi*

smalt map [**MAP-OPTIONS**] *INDEX READ-FILE* [*MATE-FILE*]
: loads the index into memory and aligns single or (if *MATE-FILE* is specified) paired-end reads against the reference sequences.

smalt version
: prints version information.

smalt help
: for a brief summmary of this software.

Help on individual tasks

smalt TASK -H
: e.g. *smalt index -H* for help on options influencing the generation of the hash index.

2 Description

Running *SMALT* involves two steps. First, an index of short words has to be built (smalt index). Then the sequencing reads are mapped onto the reference (smalt map). Sequence input files must be provided in FASTA or FASTQ file formats (ASCII text, see below).

SMALT uses a hash table of fixed-length words sampled along the genomic reference sequence in the file *REFSEQ-FILE* at equidistant steps. The sequencing reads in the file *READ-FILE* and, if paired-end reads are mapped, *MATE-FILE* are then mapped against the genomic reference sequences one-by-one. The sequence files *REFSEQ-FILE*, *READ-FILE* and *MATE-FILE* have to be in FASTA or FASTQ format. First, exactly matching seeds are identified in the reference sequences by looking up the *k*-mer words of the read in the hash index. Based on these seeds, potentially matching sequence segments are selected for alignment by a Smith-Waterman algorithm.

3 Options

3.1 INDEX-OPTIONS

-k ***wordlen*** Sets the length of the hashed words. *wordlen* is an integer with 2 < *wordlen* <= 20 (default: 13).

-s ***skipstep*** Sampling step size, i.e. the distance between successive words that are hashed along the genomic reference sequence. With the option **-s** 1 every word is hashed, with **-s** 2 every second word, with **-s** 3 very third etc. By default *skipstep* is set equal to *wordlen.*

3.2 MAP-OPTIONS

-a When this flag is set, explicit alignments are output along with the mappings.

-c ***mincover*** Only consider mappings where the k-mer word seeds cover the query read to a minimum extent. If *mincover* is an integer or floating point value > 1.0, at least this many bases of the read must be covered by *k*-mer word seeds. If *mincover* is a floating point value <= 1.0, it specifies the fraction of the query read length that must be covered by k-mer word seeds. This option can be used only when the '-x' flag is also set.

-d ***scorediff*** Set a threshold of the Smith-Waterman alignment score relative to the maximum score. This option affects the way alignments are reported in single-read mode. For each read all alignments resulting in Smith-Waterman scores within *scorediff* of the maximum score are reported. Mappings with scores lower than this value are skipped. If *scordiff* (an integer) is set to a value < 0, all alignments are reported with scores above the threshold set by the **-m** *minscor* option.
If set to 0 (default) only mappings with the best score are reported. Reads

with multiple best mappings are reported as unmapped. This is also how read pairs are reported irrespective of the value of *scorediff*.

-f *format* Specifies the output format. *format* can be one of the following strings:

cigar (default) Compact Idiosyncratic Gapped Alignment Report (see `http://www.sanger.ac.uk/resources/software/ssaha2`)

sam Sequence Alignment/Map format (`http://samtools.sourceforge.net`) with hard clipped sequences.

samsoft like *sam* but using soft clipping

ssaha native output format of the SSAHA2 software package (`http://www.sanger.ac.uk/resources/software/ssaha2`)

-H Print instructions on screen.

-i *insertmax* Maximum insert size for paired-end reads. *insertmax* is a positive integer (default 500).

-j *insertmin* Minimum insert size for paired-end reads *insertmax* is a positive integer (default 0).

-m *minscor* Sets an absolute threshold of the Smith-Waterman scores. Mappings with scores below that threshold will not be reported. *minscor* is a positive integer (default *minscor* = *wordlen* + *skipstep* - 1).

-n *nthreads* Run *SMALT* using multiple threads. *nthread* is the number of threads forked including the main thread. A maximum of 8 threads can be forked.

-o *oufilnam* Write mapping output (e.g. SAM lines) to a separate file named *oufilnam*. If this option is not specified, mappings are written to standard output together with other messages.

-p Report partial alignments if they are complementary on the query read (split or chimeric reads). A maximum of two partial alignments are output per read. The second alignment is labelled 'P' ('-f ssaha' or '-f cigar' formats) or has the 'scondary alignment' bit-flag (0x100) of the SAM FLAG field raised ('-f sam' or '-f samsoft').

-q *minbasq* Sets a base quality threshold 0 <= *minbasq* <= 10 (default *minbasq* = 0). k-mer words of the read with base pairs that have a base quality below this threshold are not looked up in the hash index.

-r *seed* If the there are multiple mappings with the same best alignment score report one one picked at random. This is relevant only in paired-end mode or with the option **-d** 0 (the default). *seed* is a positive integer used to to seed the pseudo-random generator. With *seed* = 0 a seed is derrived from the current calendar time. Without this option (default) reads with multiple best mappings are reported as unmapped.

-y *minid* Filters output alignment by a threshold in the number of exactly matching nucleotides. *minid* is a positive integer or a floating point number <= 1.0 specifying a fraction of the read length.

-x This flag triggers a more exhaustive search for alignments at the cost of decreased speed. In paired-end mode each mate is mapped independently. (By default the mate with fewer hits in the hash index is mapped first and the vicinity is searched for its mate.)

-w Output complexity weighted Smith-Waterman scores.

4 Note on Paired-End Reads

The two mates of paired reads are expected in separate FASTQ input files. The mates of a pair are identified by the position in the respective FASTQ file. Read names are *not* checked by the software. It is up to the user to make sure the mates of the i-th read are the i-th sequences in the FASTQ files. As a consequence paired and unpaired reads cannot be mixed in the input files.

The **-i** and **-j** options specify the expected insert size range influence the way in which the mates are aligned. The mate with fewer hits in the hash index is mapped first and the vicinity defined by the expected range is searched for its mate. If the **-x** option is specified both mates are aligned independently.

In some output formats the reads are labelled or flagged, e.g. as a *proper* pair in the SAM format, with respect to the insert size range. But the alignments of all mates will be reported regardless of the range specfied with the **-i** and **-j** options.

4.1 Definition of 'proper' pairs in output formats

The mates of a read pair are in *proper* orientation when they map to opposite strands with the 5'-ends on the outside of the double stranded segment spanned by the pair as expected from the Illumina paired-end (PE) libraries with short insert lengths. A *proper* pair has both mates mapped in *proper* orientation within the expected insert range (specified with the **-i** and **-j** options).

Paired reads from other library preparation protocols, like the Illumina mate-pair libraries for long insert lenghts, or from other sequencing platforms would have to be converted to input files expected from the Illumina PE format. This is important for correct labelling, e.g. of 'proper' pairs, and also for assigning the correct mapping quality scores.

4.2 Extension of the CIGAR output format

The CIGAR output format (option **-f** *cigar*) produced by *smalt* comprises a label after the GIGAR tag (e.g. label A in GIGAR:A:51). The labels have the following meaning:

A mates are in *proper* orientation within the limits specified by the **-i** and **-j** options.

B mates in *proper* orientation outside the limits specified by the **-i** and **-j** options but on the same reference sequence (i.e. chromosome or contig).

C mates are not in *proper* orientations but on the same chromosome or contig.

D mates are mapped to different chromosomes or contigs.

N read could not be mapped.

P Alignment is the second partial alignment of a split (chimaeric) read (only with **-p** flag).

S Read was mapped as a single read (sole mapped read of a pair).

5 Note on Smith-Waterman Scores

SMALT uses 'standard' Smith-Waterman scores:
match: +1; mismatch: -2; gap opening: -4; gap extension: -3.
There is currently no way to modify these settings.

The options **-d** *scordiff* and **-m** *minscor* which determine how many alignments are reported, are based on Smith-Waterman scores rather than e.g. edit distance. Calculating *scordiff* from the edit distance is simple if *SMALT* is run without the **-w** flag.

6 Memory Requirements

The memory footprint of *SMALT* is determined primarily by the total number N of base pairs of the genomic reference sequences and by the word length k (option **-k** k) and the sampling step s (option **-s** s) with which the hash index is generated. *SMALT* requires approx. $4 * (4^k + N/s)$ or $12 * N/s$ (whichever number is smaller) bytes of memory for the index. The genomic reference sequences occupy about $2N/5$ bytes during construction of the index and during mapping.

For example constructing an index of words of length 13 sampled at every 6^{th} position (options **-k** 13 **-s** 6) for the human genome ($N = 3x10^9$) requires 4 GB. Mapping reads with this index requires 3.3 GB of memory. An index of the human genome built with options **-k** 13 **-s** 13 (default) requires 4.3 GB during construction and 2.3 GB during mapping. The recommended setting for 100 bp Illumina reads, **-k** 20 **-s** 13, requires 4.0 GB for construction and 3.8 GB for mapping.

7 Index Files

The command
smalt index [**-k** *k*] [**-s** *s*]*INDEX REFSEQ-FILE*
writes 2 files to disk:

`INDEX.sma` Compressed set of reference sequences for which the hash table of k-mer words was generated. $N * 2/5$ bytes where N is the total number of base pairs of the genomic reference sequences.

`INDEX.smi` The actual hash index. The file size is about $\min(4 * (4^k + N/s), 12 * N/s)$ bytes.

8 Sequence File Formats

Sequence input files are expected in FASTA or FASTQ format (see `http://en.wikipedia.org/wiki/FASTQ_format`).
Variations of the FASTQ format are explained in `http://maq.sourceforge.net/fastq.shtml`.

9 Version

Version: 0.5.5 of July 5, 2011.

10 License and Copyright

11 Authors

SMALT was written by Hannes Ponstigl [`hp3@sanger.ac.uk`] at the Wellcome Trust Sanger Institute, Cambridge, UK in 2010.

12 Examples

12.1 Paired-end Illumina-Solexa reads, human genome

12.1.1 Longer reads (≈ 100 bp)

The insert size be 300bp, the reads provided in two FASTQ files: `mate1.fq` contains the 1^{st} and `mate2.fq` the 2^{nd} read of each pair. The human chromosome sequences be in the FASTA file `GRCh37.fa`.

Build the hash index: `smalt index -k 20 -s 13 hs37k20s13 GRCh37.fa`
This writes an index file `hs37k20s13.smi` of 2.7 GB and a sequence file `hs37k20s13.sma` of 1.2 GB to the disk using 4.0 GB of memory.

Map the reads: `smalt map -f samsoft -o hs37map.sam hs37k13s13 mate1.fq mate2.fq`

This writes a file `hs37map.sam` with alignments in SAM format using 'soft clipping' which retains the entire read sequence.

12.1.2 short reads (36 bp)

Because of the short read length, the hash index should be built with a smaller sampling step size, for example **-s** 3 or **-s** 2. Larger step sizes would result in reduced sensitivity and increased error rate.

Build the hash index: `smalt index -s 3 hs37k13s3 GRCh37.fa`

This writes and index file `hs37k13s13.smi` of 3.8 GB and a sequence file `hs37k13s13.sma` of 1.2 GB to the disk. The memory footprint is 5.3 MB.

Map the reads: `smalt map -o hs37map.cig hs36k13s3 mate1.fq mate2.fq`

This writes a file `hs37map.cig` with mappings in CIGAR format.

12.2 Single Roche-454 reads (human)

Build the hash index: `smalt index -s 4 hs37k13s4 GRCh37.fa`

This writes an index file `hs37k13s4.smi` of 3.0 GB and a sequence file `hs37k13s4.sma` of 1.1 GB to the disk using 4.4 GB of memory.

Map the reads: `smalt map -f ssaha -o hs37map.ssaha hs36k13s4 reads.fq`

This writes a file `hs37map.ssaha` with alignments in SSAHA2 native format.

12.3 Single Illumina-Solexa reads (bacterial genome)

Map 76 bp single reads (FASTQ file `reads.fq` of the bacterium S. suis (FASTA file `suis.fa`).

For small genomes, one can often afford using the most sensitive settings for the hash index, i.e. **-s** 1, and possibly reduce the word length, e.g. **-k** 11.

Build the hash index: `smalt index -k 11 -s 1 suisk11s1 suis.fa`

Map the reads: `smalt map suisk11s1 reads.fq`

This writes a CIGAR lines to standard output.

13 Tuning performance

By tuning two parameters, the word length (**-k** *wordlen*) and the step size (**-s** *stepsiz*) with which the index is built, one can trade sensitivity and accuracy against speed and memory efficiency.

platform	genome	length	options	memory
Illumina	*H. sapiens*	108 bp	-k 20 -s 13	3.8 GB
Illumina	*H. sapiens*	76 bp	-k 13 -s 6	3.3 GB
Illumina	*H. sapiens*	54 bp	-k 13 -s 4	4.3 GB
Illumina	*H. sapiens*	36 bp	-k 13 -s 3	5.3 GB
Illumina	*C. elegans*	76 bp	-k 13 -s 4	
Illumina	*P. falciparum*	72 bp	-k 13 -s 2	
Illumina	*C. suis*	72 bp	-k 13 -s 2	
Roche-454	*H. sapiens*	200 bp	-k 13 -s 4	4.3 GB
Capillary	*H. sapiens*	500 bp	-k 13 -s 4	4.3 GB
small RNAs	*H. sapiens*	$\geq$ 12 bp	-k 11 -s 2	7.3 GB

A necessary condition for a read to register a match on a segment of the genomic reference is that there be at least one contiguous stretch of *wordlen* identical nucleotides between the two sequences.

This is, however, not a sufficient criterion. Because hashed words are sampled only every *stepsiz* base pairs along the reference, any particular word in the sequencing read may be missed. But even when there is a contiguous segment of $wordlen + stepsiz - 1$ identical nucleotide between the sequences, a match may be missed on rare occasions because, depending on the command line flags, heuristics are employed to speed up program execution.

Generally *wordlen* should be set to 13 (the default) but can be increased to 20 for Illumina reads of $>$ 100bp length or reduced to 11 for very short query sequences of 11 – 24 base pairs. The choice of *stepsiz* is more critical and depends on the available computer memory, on the size of the genome and the expected degree of variation between the sequenced genome and the reference, as well as on the sequencing platform with its inherent sequencing error profile.

The table is intended as a guideline for economical choices of **-s** *stepsiz* for a range of scenarios. Reducing *stepsiz* results in lower error rates at the cost of a reciprocal increase of the memory footprint and, particularly with the **-x** flag, of reduced execution speeds.