SMALT Manual

September 1, 2010
Version 0.3.2

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).

SMALT uses a hash table of fixed-length words sampled along the genomic reference sequence in the FASTA file *REFSEQ-FILE* at equidistant steps. The sequencing reads in the FASTQ files *READ-FILE* and, if paired-end reads are mapped *MATE-FILE* are then mapped against the genomic reference sequences one-by-one. First, exactly matching seeds are identified in the reference sequences by looking up the *k*-mer words of the read in the hash table. 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* <= 15 (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.

-d *scordiff* Set a threshold of the Smith-Waterman alignment score relative to the maximum score. All mappings resulting in Smith-Waterman scores within *scorediff* of the maximum score are reported for each read. Mappings with scores lower than this value are skipped.

scorediff is an integer. If set to 0 (default) only mappings with the best score are reported. If set to a value ¡ 0, report all alignments with scores above the threshold set by the **-m** *minscor* option.

-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 equals *wordlen*).

-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 complement each other on the query read (split or chimeric reads).

-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 the mate.)

-w Output complexity weighted Smith-Waterman scores.

4 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 $4 * (4^k + N/s)$ bytes of memory for the index. In addition, the genomic reference sequences occupy N bytes during construction of the index and $N/3$ bytes 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 5.3

GB. Mapping reads with this index requires 3.3 GB of memory. An index built with options **-k** 13 **-s** 13 (default) requires 4.3 GB during construction and 2.3 GB during mapping.

5 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. $N/3$ 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 $4 * (4^k + N/s)$ bytes.

6 Version

Version: 0.3.2 of September 1, 2010.

7 License and Copyright

License Binaries are available free of charge. The source code will be made available shortly under the GNU General Public License (`http://www.gnu.org/licenses/`).

8 Authors

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

9 Examples

9.1 Paired-end Illumina-Solexa reads, human genome

9.1.1 Longer reads (≈ 100 bp)

The insert size is 300bp. The reads are 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 are in the FASTA file `NCBI37.fa`.

Build the hash index: `smalt index hs37k13s13 NCBI37.fa`
This writes the index file `hs37k13s13.smi` and the sequence file `hs37k13s13.sma` to the disk.

Map the reads: `smalt map -f samsoft -o hs37mappings.sam hs36k13s13 mate1.fq mate2.fq`

This writes a file `hs37mappings.sam` with mappings in SAM format.

9.1.2 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 NCBI37.fa`

This writes and index file `hs37k13s13.smi` of 4.5 GB and a sequence file `hs37k13s13.sma` of 1.1 GB to the disk.

Map the reads: `smalt map -o hs37mappings.cig hs36k13s3 mate1.fq mate2.fq`

This writes a file `hs37mappings.cig` with mappings in CIGAR format.

9.2 Single Roche-454 reads (human)

Build the hash index: `smalt index -s 4 hs37k13s4 NCBI37.fa`

This writes and index file `hs37k13s13.smi` of 3.4 GB and a sequence file `hs37k13s13.sma` of 1.1 GB to the disk.

Map the reads: `smalt map -f ssaha -o hs37mappings.ssaha hs36k13s3 reads.fq`

This writes a file `hs37mappings.ssaha` with mappings in SSAHA2 native format.

9.3 Single Illumina-Solexa reads (bacterial)

Map 75 bp single reads (FASTQ file `reads.fq` of the bacterium S. suis (FASTA file `suis.fa`).

For small genomes, one can almost always afford to use 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 suisk13s1 suis.fa`

Map the reads: `smalt map suisk13s1 reads.fq`

This writes a CIGAR lines to standard output.

10 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 on the one hand against speed and memory efficiency on the other.

platform	genome	length	options	memory
Illumina	*H. sapiens*	100 bp	-k 13 -s 6	3.3 GB
Illumina	*H. sapiens*	72 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*	72 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*	$\geq$ 12bp	-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 reduced to 11 for very short query sequences of 11 – 24 base pairs. The choice of *stepsiz* is far more critical and depends on the available computer memory, on the size of the genome and its variational distance to the reference, as well as on the sequencing platform with its inherent sequencing error profile.

The following 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.