GenomeMapper Manual

version 0.4.1s

Contents

GenomeMapper is a fast and sensitive alignment tool for short sequence reads generated on the Illumina Genome Analyzer platform. It aligns short reads against a reference sequence allowing for a user-defined number of mismatches and gaps. For this reason the user has full control over runtime and sensitivity of GenomeMapper. Based on a hash index, GenomeMapper follows a seed-and-extend approach supplemented by k-banded alignments (similar to Needleman-Wunsch) for accurate and consistent read alignments.

GenomeMapper has been incorporated into the Illumina Sequencing project software suite SHORE (http://1001genomes.org).

GenomeMapper exhibits high versatility:

- Short reads can be provided in three different file formats. FASTA, FASTQ and SHORE flat file format (see pg. 12) will be automatically identified by GenomeMapper.
- GenomeMapper allows for an arbitrary number of mismatches and gaps up to a maximum of 10.
- Hash index sizes (also called seed lengths) range from 5 to 13. This allows for a multitude of different applications, including read mapping to distantly related

genomes, mapping longer reads from other Next Generation Sequencers, sRNA, mRNA and ChipSeq reads and so on.

- By default GenomeMapper reports all best hits for a given read, though command line option can be used to optain all hits, though at the costs of additional runtime. Additionally, GenomeMapper allows to randomly assign or drop repetitive reads.
- GenomeMapper can run on several cores in parallel.
- GenomeMapper's of mismatch- and gapscores can be adjusted. However, suitable default values facilitate a simple usage.
- GenomeMapper supports two output file formats: BED format and SHORE format (pg. 12), which includes an human readable alignment.

System requirements

GenomeMapper is written in C and was developed on Linux. It merely requires standard C libraries and the C compiler gcc.

GenomeMapper has varying memory and disk space requirements, depended on the reference sequence and the index size. For example, building an index with a hash size of 12 of the human genome, GenomeMapper requires ∼15.8 GB of disk space and ∼18.5 GB RAM during execution of `genomemapper`. With the same hash size, the index of *Arabidopsis thaliana* (120 Mbp) needs ∼1.9 GB of disk space and ∼3.8 GB RAM.

If you are interested, you can approximate the RAM requirements as

$$(768 \cdot r) + \left(\frac{g + n \cdot r \cdot 4 + l \cdot p + (7 + 2 \cdot p) \cdot c}{1.048.576} \right) \text{MB}$$

of memory for `genomemapper`, where:
r is 2 if both indexes (forward and reverse) are compiled and 1 if only the forward index will be created,
g is the total size of the reference sequence in base pairs,
n is the number of occurrences of all seeds of the reads in the genome; this value is promted by `gmindex` if the verbose mode is enabled,
l is the length of the longest chromosome/contig in the reference sequence, against measured in bp,
p is the pointer size in byte (for 32-bit system is this 4 bytes, for 64-bit 8 bytes),
c is the hit container size (see option `-c`, pg. 11).

Installation

Download GenomeMapper at http://1001genomes.org/downloads/genomemapper.html. Change your working directory to the folder and type:

for Linux:

```
tar -zxf genomemapper.tar.gz
cd genomemapper
make
```

for MAC (64-bit OS):

```
tar -zxf genomemapper.tar.gz
cd genomemapper
mv Makefile.MAC64 Makefile
make
```

The package now contains two programs, `gmindex` and `genomemapper`. `gmindex` creates the hash index. `genomemapper` performs the read mapping.

Quick Guide

To build the (forward and reverse) index of the reference file `genome.fa` with seed-length (also called hash size) of 11, type:

```
$genomemapper-path$ ./gmindex -i genome.fa -s 11
```

To map the query file `reads.fl` against the reference `genome.fa` allowing for 4 mismatches and 2 gaps (with a total of 4 edit operations), type:

```
$genomemapper-path$ ./genomemapper -i genome.fa -q reads.fl -M 4 -G 2 -E 4
```

To print the mapped reads in `mapped_reads.fl` in SHORE's flat file output format and the unmapped ones in `unmapped_reads.fl` type:

```
$genomemapper-path$ ./genomemapper -i genome.fa -q reads.fl
-M 4 -G 2 -E 4 -o mapped_reads.fl -u unmapped_reads.fl
```

To use different file formats than the SHORE formats, only the output file format has to be specified, the input format will be automatically identified. Note that the unmapped read output format is always the read input format.

The pre-processing program `gmindex`

The program `gmindex` constitutes the pre-processing step of GenomeMapper. It translates the reference sequence into a hash index to the same directory. Brief summary:

```
./gmindex [options] -i <genome file>
```

Parameters (bold ones are mandatory):

-i <genome fasta file>	Specifies the input file containing the reference sequences in multi FASTA format. The output files will be written in the same directory.
-s <int>	Specifies the hash size (also called seed length). The allowed seed lengths are from 5 to 13. Default seed length is 12.
-r	Disables the index creation of the reverse strand. By default both indices of forward and reverse strand are generated.
-v	Set to verbose mode.

Mandatory parameter

-i <genome fasta file>

The genome file is a multiple FASTA file containing the reference sequences. The index files are stored in the same directory as the reference file. Their file names consist of the reference file name followed by differing suffixes. It is essential that the reference sequence file and the index file reside in the same directory. The maximal length of the FASTA ids is 50. Within the sequences lower case bases are not considered as masked. Any empty lines or lines containing only whitespace characters are ignored.

All IUPAC characters unequal to the four bases A, C, G and T are considered as mismatches against every other symbol.

The genome size, i.e. the sum of the lengths of all chromosomes/contigs, must not exceed $2^{32} - n$ bp with n chromosomes/contigs, where $n < 2^{24}$.

To avoid long computation times due to repeats, repeat masking before using `gmindex` is recommended.

Optional parameters

-s <seed length, or hash size>

Sensitivity of the mapping depends on the seed length. Simple seed strategies require an *identical* substring between read and reference sequence of at least seed length. The longer the seed, the lower the runtime of `genomemapper`, but also the lower the sensitivity.

The seed length can be adjusted somewhere between 5 and 13. Default is 12. Note that mapping millions of reads with `genomemapper` with very small seed lengths (< 8) can take days.

-r

By default, `gmindex` builds two indices, one for the forward and the other one for the reverse strand. -r will switch off the reverse strand. `genomemapper` won't perform any mappings on the reverse strand.

-v

Set to verbose mode.

The mapping program `genomemapper`

`genomemapper` performs the mapping of short reads against a single reference sequence.

```
./genomemapper [options] -i <genome file>
                         -q <query file>
```

Parameters (bold ones are mandatory):

-i <genome fasta file>	Specifies the reference file.
-q <query file>	Specifies the input file containing the reads which should be mapped against the reference. FASTA, FASTQ and SHORE flat file format are accepted (see pg. 12).
-o <output file>	Specifies the output file. Default is stdout.
-f <format>	Specifies the output file format. <format> has to be either "shore" or "bed". Default is "shore".
-u <unmapped reads file>	Specifies the output file containing all reads which could not be mapped on the reference.
-M <int>	Specifies the maximal number of allowed mismatches per alignment. Default is 3.
-G <int>	Specifies the maximal number of allowed gapped positions in the read or reference per alignment. Default is 1.
-E <int>	Specifies the maximal number of allowed edit operations per alignment, i.e. the number of mismatches plus the number of gaps must not exceed this value. Default is 3.
-t <int>	Specifies the number of threads (1 refers to the unparalized version). In order to use and see this option you need to recompile GenomeMapper with a modified Makefile. See the detailed explaination of -t for more information.
-r	Disables the mapping of the reads against the complementary reverse strand of the reference. Default is to map against both strands.
-k <int>	Limits the length of the reads to this value.
-a	Enables all-hit strategy. All alignments of a given read fulfilling the mapping criteria of `-E`, `-M` and `-G` will be reported. Default is a best-hit strategy. Attention, this strategy increases runtime drastically.
-b	Enables second-best-hit strategy. Default is a best-hit strategy. Attention, this strategy increases runtime.
-n <int>	Specifies the maximal number of randomly selected best scored alignments per read. This option works only in combination with best-hit strategy. Reporting all best hits is the default setting.

-s <int>	Only use reads which produced less seeds than this threshold. The lower this value, the faster the mapping, and the lower sensitivity. (default: off)
-x/y <int>	Compare y following characters of a seed exceeding x occurrences on the genome.
-m <int>	Specifies the penalty score of a mismatch in the alignment. It must be a positive integer < 256. Default is 4.
-g <int>	Specifies the penalty score of a gap in the alignment process. It must be a positive integer < 256. Default is 5.
-e	`genomemapper` will report the number of edit operations needed to align the read instead of the alignment score.
-d	`genomemapper` will place gaps most right in gapped alignments. Default is most left.
-h	`genomemapper` will perform Needleman-Wunsch alignments on the complete read sequence. With `-h`, slightly higher sensitivity can be achieved.
-l <int>	Increases the seed length. To speed up the mapping process the seed length can be increased compared to the index file. Larger values decrease sensitivity, but speed up the mapping. Default is the index seed length.
-w	Softens stringent gap limit. If the best alignment contains more gaps than specified by `-G` or `-E`, it will be reported, nevertheless.
-c <int>	Specifies the number of seed hits that can be stored per read. Default and minimum value is 15.000.000.
-v	Set to verbose mode.

Mandatory parameters

Input data

-i <genome fasta file>

The reference file, used also for `gmindex`.

-q <query file>

The query file contains the reads which will be aligned against the reference. The file format can either be multiple FASTA, FASTQ[1] or SHORE flat file format, which will

[1] FASTQ file format specification can be found at http://maq.sourceforge.net/fastq.shtml

be described in the section "File format specifications" on page 12. As for the genomic file, only IUPAC characters are accepted and non-base symbols count as mismatches.

To avoid long computation times due to repetitive reads, (quality and/or low complexity) pre-filtering the reads is recommended.

Any empty lines or lines containing only whitespace characters are ignored.

Optional parameters

Output files

-o <output file>

Output file in the format specified by the `-f` option.

-f shore or *-f bed*

Output file format. Either SHORE or BED file format. For a detailed explanation of both formats refer to the section "File format specifications" on page 12. BED formatted files can only report one alignment.

Default is SHORE format including an human readable alignment.

-u <unmapped reads file>

The reads which could not be mapped will be reported in this file in the input file format.

Options

-r

`genomemapper` will not read in the index of the complementary reverse strand of the reference and thus, cannot report mappings onto the minus strand.

-k <int>

Uses only the first k bases of every read.

-a

This flag switches from best-hit strategy to all-hit strategy. Not only the hits with best scores are reported, but every hit fulfilling the mapping criteria specified with `-E`, `-M` and `-G`. In this mode, `genomemapper` omits a major speedup step of the best-hit mode. Since the all-hit strategy performs more alignments than the best-hit strategy, it is slower and can be unfeasible slow for huge reference sets or small seed lengths.

Default is best-hit strategy.

-b

Activating this flag, `genomemapper` will run in second-best-hit mode: The best and the second best alignments will be reported. Since this strategy performs more alignments than the best-hit strategy, it is slower. Default is best-hit strategy.

-t <int>

Specifies the number of cores used for parallelizing GenomeMapper. Significant speed-ups are achieved for complex queries, i.e. high number of mismatches and gaps. The overhead created by the parallelization can compensate the speed-up of mappings with simpler restrictions. We recommend to try different values.

In order to use the parallelized version of GenomeMapper you need to re-compile GenomeMapper with a modified version of the Makefile. Open the Makefile with any kind of text editor and put a `"#"` at the begining of line 6. Second you need to delete the `"#"` symbol at the begining of line 8. Once this is done store this new version of the Makefile. Now type `make clean` which will remove the unparallelized version. Finally type `make` again and GenomeMapper will now allow you to adjust the number of threads it runs on.

-n <int>

The maximal number of randomly chosen hits being reported, works only with best-hit strategy. By default all alignments are reported.

-s <int>

If a seed of a read has more than s occurrences, it will be discarded. The lower this value, the faster `genomemapper` will be, but the lower is the sensitivity. This option takes no effect combined with `-a` and `-b`.

-x <int> / -y <int>

If a seed of a read has more than x occurrences, the y following characters of the seed will be compared between read and genome. The seed is only considered valid if no mismatch is found, otherwise the seed is discarded. These options take no effect combined with `-a` and `-b`.

-M <int>

This value specifies the maximal number of mismatches in the alignment [0-10]. Default is 3.

`genomemapper` can find all n-mismatch mappings for $n < \left\lfloor \frac{\text{read length}}{\text{seed length}} \right\rfloor$. Some hits can be missed where the mismatches are distributed in a way that there is no continuous stretch of identical bases which has the length of the seed length.

-G <int>

This value specifies the maximal number of gapped positions in the alignment [0-10]. Each position in a gap is counted separately. Default is 1.

Beware that the number of gaps can drastically influence the runtime.

It is recommended to set the number of gaps smaller than the number of mismatches.

-E <int>

Maximal number of edit operations per alignment [0-10]. Edit operations is defined as the number of mismatches plus the number of gaps (also called Levenshtein distance). Default is 3.

`-E` controls `-M` and `-G`: If the number of edit operations is set to a smaller value than the number of mismatches and gaps, both are adjusted to the number of edit operations.

`-E` allows for complex requests: for instance, if you want to find reads with a Levenshtein distance of 4, but you don't want to allow for 4 gaps, you can specify `-E 4`, `-M 4` and `-G 3` for 3 gaps.

-m <double>
-g <double>

The user can define the penalty scores of mismatches and gaps in the alignment process. They must be positive numbers < 256. The default setting is 4 for mismatches and 5 for gaps. That way an alignment with n mismatches is always preferred to an

alignment with n gaps and an alignment with $n-1$ gaps is preferred to an alignment with n mismatches up to $n = 4$.

Note that the score for a match is 0. For accurate alignments, the following constraint has to be fulfilled: 0 = match score < mismatch score < gap score.

-e

With this flag, `genomemapper` is instructed to print the number of edit operations instead of the alignment scores (default) in the output file.

-d

This flag causes `genomemapper` to place gaps most right in gapped alignments. Default is most left.

-h

This flag causes `genomemapper` to enforce an alignment of the whole read sequence to the reference. By default `genomemapper` aligns only sub-strings of reads which have not been used for the seed mapping. To ensure the uniqueness of the alignments gapped alignments are always performed on the whole read.

Activating `-h` will yield slightly more sensitive mappings.

-l <int>

Increases the seed length. Higher values yield lower sensitivity, but speed up the mapping process. Default setting is the hash size.

-w

The gap limit `-G` is softened with this option. In seldom cases where best alignments don't fulfill the mapping criterias worse alignments can reported instead. For instance, an alignment with 2 gaps has a better score than an alignment with 4 mismatches. But assuming that only 1 gap and 4 mismatches have been allowed, the worse alignment fulfills the mapping criteria whereas the better one does not.

Default is to uphold the stringent gap limit and not report mappings with more gaps than desired.

-c <int>

This value specifies how many seeds hits can be stored per read (seed container size). Default is 15,000,000. The default setting should work well for almost all practical cases. A warning is printed out if reads which could not have been mapped due to this limitation. These reads can be mapped in a subsequent mapping run with a higher value for option `-c`.

-v

Set to verbose mode.

File format specifications

SHORE input format

The flat file format files stores read information from Illumina GA2 sequencers and *must* be tabulator-delimited. The columns are as follows in this order:

1. Read ID.
2. Read sequence.
3. Paired-end flag. Not used by `genomemapper`, used by SHORE.
4. Quality value string 1. Assumed to have exactly the same length as the read sequence. Not used by `genomemapper`.

SHORE output format

GenomeMapper's SHORE output format stores one mapping per line. If the input read file is in FASTA format, only the columns 1 to 9 described below are reported, for FASTQ only columns 1 to 9 and 11 and in case of SHORE flat file format all 13 columns will be printed.

The columns are tab delimited.

1. Strain identifier
2. Chromosome/Contig ID as specified in the input fasta file of `gmindex` (`-i` option)
3. Left-most reference position of the alignment (positions starting from 1)

4. Alignment on the reference sequence: Matching bases are reported once, whereas mismatches and gaps are represented as two characters in squared brackets. The first character refers to the reference and the second to the read. These sequences display the positive strand of the reference. Gaps are reported as dashes ('-').
5. Read ID as specified in the query file (`-q` option)
6. Strand: 'D' for the forward, 'P' for the reverse
7. Alignment score or number of edit operations (`-e` option)
8. Number of mappings of the read
9. Read length
10. Undefined, SHORE interface.
11. Paired-End flag, SHORE interface.
12. Illumina GA2 quality, SHORE interface.
13. Sanger scaled quality, SHORE interface.
14. Chastity value, SHORE interface.

BED output format

GenomeMapper prints the first 6 columns of the BED file format. The specification can be found at http://genome.ucsc.edu/FAQ/FAQformat#format1.

1. Chromosome/Contig ID as specified in the input fasta file of `gmindex` (`-i` option)
2. Left-most reference position of the alignment (positions starting from 0)
3. The last position of the alignment within the reference. The specified position is not included in the alignment.
4. Read ID as specified in the query file (`-q` option)
5. Alignment score or number of edit operations (`-e` option)
6. Strand: '+' for the forward, '-' for the reverse

Contact

GenomeMapper was written by Korbinian Schneeberger, Jörg Hagmann and Stephan Ossowski at the Max Planck Institute for Developmental Biology, Tübingen, Germany and Sandra Gesing at the University of Tübingen, Germany, in 2008/09. We appreciate any kind of feedback. Please do not hesitate and contact us in any case of problems or questions:

korbinian.schneeberger@tuebingen.mpg.de
stephan.ossowski@tuebingen.mpg.de
joerg.hagmann@tuebingen.mpg.de