Bowtie

An ultrafast memory-efficient short read aligner

Site Map

- Home
- News archive
- Getting started
- Manual

Latest Release

Bowtie 0.12.7 9/7/10

- Please cite: Langmead B, Trapnell C, Pop M, Salzberg SL. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* 10:R25.
- For release updates, subscribe to the mailing list.

Related Tools

Crossbow: Genotyping, cloud computing
Tophat: RNA-Seq splice junction mapper
Cufflinks: Isoform assembly, quantitation
Myrna: Cloud, differential gene expression

Pre-built indexes

H. sapiens, UCSC hg18 **2.7 GB**

or: part 1 - 1.7 GB, part 2 - 1.0 GB
colorspace: full, or part 1, part 2

H. sapiens, UCSC hg19 **2.7 GB**

or: part 1 - 1.7 GB, part 2 - 1.0 GB
colorspace: full, or part 1, part 2

H. sapiens, NCBI v36 **2.7 GB**
or: part 1 - 1.7 GB, part 2 - 1.0 GB
colorspace: full, or part 1, part 2

H. sapiens, NCBI v37 **2.7 GB**
or: part 1 - 1.7 GB, part 2 - 1.0 GB
colorspace: full, or part 1, part 2

M. musculus, UCSC mm8 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

M. musculus, UCSC mm9 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

M. musculus, NCBI v37 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

R. norvegicus, UCSC rn4 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

B. taurus, UMD v3.0 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

C. familiaris, UCSC canFam2 **2.4 GB**
or: part 1 - 1.5 GB, part 2 - 900 MB
colorspace: full, or part 1, part 2

G. gallus, UCSC, galGal3 1018 MB
colorspace: full

D. melanogaster, Flybase, r5.22 150 MB
colorspace: full

A. thaliana, TAIR, TAIR9 120 MB
colorspace: full

C. elegans, Wormbase, WS200 75 MB

colorspace: full

S. cerevisiae, CYGD 15 MB

colorspace: full

E. coli, NCBI, st. 536 5 MB

colorspace: full

All indexes are for assemblies, not contigs. Unplaced or unlocalized sequences and alternate haplotype assemblies are excluded.

Some unzip programs cannot handle archives >2 GB. If you have problems downloading or unzipping a >2 GB index, try downloading in two parts.

Check .zip file integrity with MD5s.

Pre-built indexes are compatible with Bowtie versions 0.9.8 and later. For older indexes, please contact us.

Publications

- Langmead B, Trapnell C, Pop M, Salzberg SL. **Ultrafast and memory-efficient alignment of short DNA sequences to the human genome**. *Genome Biology* **10**:R25.
- Langmead B, Schatz M, Lin J, Pop M, Salzberg SL. **Searching for SNPs with cloud computing**. *Genome Biology* **10**:R134.
- Trapnell C, Pachter L, Salzberg SL, **TopHat: discovering splice junctions with RNA-Seq**. *Bioinformatics* 2009 25(9):1105-1111.

Other Documentation

- AGBT poster, 2/09 (.ppt, .pdf)
- NCBI Presentation, 11/08 (.ppt, .pdf)
- UMD Biosciences poster, 11/08 (.ppt, .pdf)

Authors

- Ben Langmead
- Cole Trapnell

Links

- Bowtie sourceforge.net project
  - Request a feature
  - Report a bug
  - Mailing list
- SAMtools
- SEQanswers
- University of Maryland

- UMD CBCB
- UMD Computer Science
- UMIACS
- J.H. Bloomberg School of Public Health
- JHSPH Biostatistics

Table of Contents

Bowtie 0.12.7

What is Bowtie?

Bowtie is an ultrafast, memory-efficient short read aligner geared toward quickly aligning large sets of short DNA sequences (reads) to large genomes. It aligns 35-base-pair reads to the human genome at a rate of 25 million reads per hour on a typical workstation. Bowtie indexes the genome with a Burrows-Wheeler index to keep its memory footprint small: for the human genome, the index is typically about 2.2 GB (for unpaired alignment) or 2.9 GB (for paired-end or colorspace alignment). Multiple processors can be used simultaneously to achieve greater alignment speed. Bowtie can also output alignments in the standard SAM format, allowing Bowtie to interoperate with other tools supporting SAM, including the SAMtools consensus, SNP, and indel callers. Bowtie runs on the command line under Windows, Mac OS X, Linux, and Solaris.

Bowtie also forms the basis for other tools, including TopHat: a fast splice junction mapper for RNA-seq reads, Cufflinks: a tool for transcriptome assembly and isoform quantitiation from RNA-seq reads, Crossbow: a cloud-computing software tool for large-scale resequencing data,and Myrna: a cloud computing tool for calculating differential gene expression in large RNA-seq datasets.

If you use Bowtie for your published research, please cite the Bowtie paper.

What isn't Bowtie?

Bowtie is not a general-purpose alignment tool like MUMmer, BLAST or

Vmatch. Bowtie works best when aligning short reads to large genomes, though it supports arbitrarily small reference sequences (e.g. amplicons) and reads as long as 1024 bases. Bowtie is designed to be extremely fast for sets of short reads where (a) many of the reads have at least one good, valid alignment, (b) many of the reads are relatively high-quality, and (c) the number of alignments reported per read is small (close to 1).

Bowtie does not yet report gapped alignments; this is future work.

Obtaining Bowtie

You may download either Bowtie sources or binaries for your platform from the Download section of the Sourceforge project site. Binaries are currently available for Intel architectures (`i386` and `x86_64`) running Linux, Windows, and Mac OS X.

Building from source

Building Bowtie from source requires a GNU-like environment that includes GCC, GNU Make and other basics. It should be possible to build Bowtie on a vanilla Linux or Mac installation. Bowtie can also be built on Windows using Cygwin or MinGW. We recommend TDM's MinGW Build. If using MinGW, you must also have MSYS installed.

To build Bowtie, extract the sources, change to the extracted directory, and run GNU `make` (usually with the command `make`, but sometimes with `gmake`) with no arguments. If building with MinGW, run `make` from the MSYS command line.

To support the `-p` (multithreading) option, Bowtie needs the `pthreads` library. To compile Bowtie without `pthreads` (which disables `-p`), use `make BOWTIE_PTHREADS=0`.

The `bowtie` aligner

`bowtie` takes an index and a set of reads as input and outputs a list of alignments. Alignments are selected according to a combination of the `-v`/`-n`/`-e`/`-l` options (plus the `-I`/`-X`/`--fr`/`--rf`/ `--ff` options for paired-end alignment), which define which alignments are legal, and the `-k`/`-a`/`-m`/`-M`/`--best`/`--strata` options which define which and how many legal alignments should be reported.

By default, Bowtie enforces an alignment policy similar to Maq's default quality-aware policy (`-n` 2 `-l` 28 `-e` 70). See the -n alignment mode section of

the manual for details about this mode. But Bowtie can also enforce a simpler end-to-end k-difference policy (e.g. with `-v` 2). See the -v alignment mode section of the manual for details about that mode. [The -n alignment mode] and the -v alignment mode are mutually exclusive.

Bowtie works best when aligning short reads to large genomes (e.g. human or mouse), though it supports arbitrarily small reference sequences and reads as long as 1024 bases. Bowtie is designed to be very fast for sets of short reads where a) many reads have at least one good, valid alignment, b) many reads are relatively high-quality, c) the number of alignments reported per read is small (close to 1). These criteria are generally satisfied in the context of modern short-read analyses such as RNA-seq, ChIP-seq, other types of -seq, and mammalian resequencing. You may observe longer running times in other research contexts.

If `bowtie` is too slow for your application, try some of the performance-tuning hints described in the [Performance Tuning] section below.

Alignments involving one or more ambiguous reference characters (`N`, `-`, `R`, `Y`, etc.) are considered invalid by Bowtie. This is true only for ambiguous characters in the reference; alignments involving ambiguous characters in the read are legal, subject to the alignment policy. Ambiguous characters in the read mismatch all other characters. Alignments that "fall off" the reference sequence are not considered valid.

The process by which `bowtie` chooses an alignment to report is randomized in order to avoid "mapping bias" - the phenomenon whereby an aligner systematically fails to report a particular class of good alignments, causing spurious "holes" in the comparative assembly. Whenever `bowtie` reports a subset of the valid alignments that exist, it makes an effort to sample them randomly. This randomness flows from a simple seeded pseudo-random number generator and is deterministic in the sense that Bowtie will always produce the same results for the same read when run with the same initial "seed" value (see `--seed` option).

In the default mode, `bowtie` can exhibit strand bias. Strand bias occurs when input reference and reads are such that (a) some reads align equally well to sites on the forward and reverse strands of the reference, and (b) the number of such sites on one strand is different from the number on the other strand. When this happens for a given read, `bowtie` effectively chooses one strand or the other with 50% probability, then reports a randomly-selected alignment for that read from among the sites on the selected strand. This tends to overassign alignments to the sites on the strand with fewer sites and underassign to sites on the strand with more sites. The effect is mitigated,

though it may not be eliminated, when reads are longer or when paired-end reads are used. Running Bowtie in `--best` mode eliminates strand bias by forcing Bowtie to select one strand or the other with a probability that is proportional to the number of best sites on the strand.

Gapped alignments are not currently supported, but support is planned for a future release.

The `-n` alignment mode

When the `-n` option is specified (which is the default), `bowtie` determines which alignments are valid according to the following policy, which is similar to Maq's default policy.

1. Alignments may have no more than `N` mismatches (where `N` is a number 0-3, set with `-n`) in the first `L` bases (where `L` is a number 5 or greater, set with `-l`) on the high-quality (left) end of the read. The first `L` bases are called the "seed".

2. The sum of the Phred quality values at *all* mismatched positions (not just in the seed) may not exceed `E` (set with `-e`). Where qualities are unavailable (e.g. if the reads are from a FASTA file), the Phred quality defaults to 40.

The `-n` option is mutually exclusive with the `-v` option.

If there are many possible alignments satisfying these criteria, Bowtie gives preference to alignments with fewer mismatches and where the sum from criterion 2 is smaller. When the `--best` option is specified, Bowtie guarantees the reported alignment(s) are "best" in terms of these criteria (criterion 1 has priority), and that the alignments are reported in best-to-worst order. Bowtie is somewhat slower when `--best` is specified.

Note that Maq internally rounds base qualities to the nearest 10 and rounds qualities greater than 30 to 30. To maintain compatibility, Bowtie does the same. Rounding can be suppressed with the `--nomaqround` option.

Bowtie is not fully sensitive in `-n` 2 and `-n` 3 modes by default. In these modes Bowtie imposes a "backtracking limit" to limit effort spent trying to find valid alignments for low-quality reads unlikely to have any. This may cause bowtie to miss some legal 2- and 3-mismatch alignments. The limit is set to a reasonable default (125 without `--best`, 800 with `--best`), but the user may decrease or increase the limit using the `--maxbts` and/or `-y` options. `-y` mode is relatively slow but guarantees full sensitivity.

The -v alignment mode

In -v mode, alignments may have no more than v mismatches, where v may be a number from 0 through 3 set using the -v option. Quality values are ignored. The -v option is mutually exclusive with the -n option.

If there are many legal alignments, Bowtie gives preference to alignments with fewer mismatches. When the --best option is specified, Bowtie guarantees the reported alignment(s) are "best" in terms of the number of mismatches, and that the alignments are reported in best-to-worst order. Bowtie is somewhat slower when --best is specified.

Strata

In the -n alignment mode, an alignment's "stratum" is defined as the number of mismatches in the "seed" region, i.e. the leftmost L bases, where L is set with the -l option. In the -v alignment mode, an alignment's stratum is defined as the total number of mismatches in the entire alignment. Some of Bowtie's options (e.g. --strata and -m use the notion of "stratum" to limit or expand the scope of reportable alignments.

Reporting Modes

With the -k, -a, -m, -M, --best and --strata options, the user can flexibily select which alignments are reported. Below we demonstrate a few ways in which these options can be combined. All examples are using the e_coli index packaged with Bowtie. The --suppress option is used to keep the output concise and some output is elided for clarity.

Example 1: -a

```
$ ./bowtie -a -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
-   gi|110640213|ref|NC_008253.1|   4930433 4:G>T,6:C>G
-   gi|110640213|ref|NC_008253.1|   905664  6:A>G,7:G>T
+   gi|110640213|ref|NC_008253.1|   1093035 2:T>G,15:A>T
```

Specifying -a instructs bowtie to report *all* valid alignments, subject to the alignment policy: -v 2. In this case, bowtie finds 5 inexact hits in the E. coli genome; 1 hit (the 2nd one listed) has 1 mismatch, and the other 4 hits have 2 mismatches. Four are on the reverse reference strand and one is on the

forward strand. Note that they are not listed in best-to-worst order.

Example 2: `-k 3`

```
$ ./bowtie -k 3 -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
-   gi|110640213|ref|NC_008253.1|   4930433 4:G>T,6:C>G
```

Specifying `-k` 3 instructs bowtie to report up to 3 valid alignments. In this case, a total of 5 valid alignments exist (see Example 1); `bowtie` reports 3 out of those 5. `-k` can be set to any integer greater than 0.

Example 3: `-k 6`

```
$ ./bowtie -k 6 -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
-   gi|110640213|ref|NC_008253.1|   4930433 4:G>T,6:C>G
-   gi|110640213|ref|NC_008253.1|   905664  6:A>G,7:G>T
+   gi|110640213|ref|NC_008253.1|   1093035 2:T>G,15:A>T
```

Specifying `-k` 6 instructs bowtie to report up to 6 valid alignments. In this case, a total of 5 valid alignments exist, so `bowtie` reports all 5.

Example 4: default (`-k 1`)

```
$ ./bowtie -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
```

Leaving the reporting options at their defaults causes `bowtie` to report the first valid alignment it encounters. Because `--best` was not specified, we are not guaranteed that bowtie will report the best alignment, and in this case it does not (the 1-mismatch alignment from the previous example would have been better). The default reporting mode is equivalent to `-k` 1.

Example 5: `-a --best`

```
$ ./bowtie -a --best -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
+   gi|110640213|ref|NC_008253.1|   1093035 2:T>G,15:A>T
-   gi|110640213|ref|NC_008253.1|   905664  6:A>G,7:G>T
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
-   gi|110640213|ref|NC_008253.1|   4930433 4:G>T,6:C>G
```

Specifying `-a` results in the same alignments being printed as if just `-a` had been specified, but they are guaranteed to be reported in best-to-worst order.

Example 6: -a --best --strata

```
$ ./bowtie -a --best --strata -v 2 --suppress 1,5,6,7 e_coli -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
```

Specifying --strata in addition to -a and --best causes bowtie to report only those alignments in the best alignment "stratum". The alignments in the best stratum are those having the least number of mismatches (or mismatches just in the "seed" portion of the alignment in the case of -n mode). Note that if --strata is specified, --best must also be specified.

Example 7: -a -m 3

```
$ ./bowtie -a -m 3 -v 2 e_coli -c ATGCATCATGCGCCAT
No alignments
```

Specifying -m 3 instructs bowtie to refrain from reporting any alignments for reads having more than 3 reportable alignments. The -m option is useful when the user would like to guarantee that reported alignments are "unique", for some definition of unique.

Example 1 showed that the read has 5 reportable alignments when -a and -v 2 are specified, so the -m 3 limit causes bowtie to output no alignments.

Example 8: -a -m 5

```
$ ./bowtie -a -m 5 -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   148810  10:A>G,13:C>G
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
-   gi|110640213|ref|NC_008253.1|   4930433 4:G>T,6:C>G
-   gi|110640213|ref|NC_008253.1|   905664  6:A>G,7:G>T
+   gi|110640213|ref|NC_008253.1|   1093035 2:T>G,15:A>T
```

Specifying -m 5 instructs bowtie to refrain from reporting any alignments for reads having more than 5 reportable alignments. Since the read has exactly 5 reportable alignments, the -m 5 limit allows bowtie to print them as usual.

Example 9: -a -m 3 --best --strata

```
$ ./bowtie -a -m 3 --best --strata -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
-   gi|110640213|ref|NC_008253.1|   2852852 8:T>A
```

Specifying -m 3 instructs bowtie to refrain from reporting any alignments for reads having more than 3 reportable alignments. As we saw in Example 6, the read has only 1 reportable alignment when -a, --best and --strata are specified,

so the -m 3 limit allows bowtie to print that alignment as usual.

Intuitively, the -m option, when combined with the --best and --strata options, guarantees a principled, though weaker form of "uniqueness." A stronger form of uniqueness is enforced when -m is specified but --best and --strata are not.

Paired-end Alignment

bowtie can align paired-end reads when properly paired read files are specified using the -1 and -2 options (for pairs of raw, FASTA, or FASTQ read files), or using the --12 option (for Tab-delimited read files). A valid paired-end alignment satisfies these criteria:

1. Both mates have a valid alignment according to the alignment policy defined by the -v/-n/-e/-l options.
2. The relative orientation and position of the mates satisfy the constraints defined by the -I/-X/--fr/--rf/--ff options.

Policies governing which paired-end alignments are reported for a given read are specified using the -k, -a and -m options as usual. The --strata and --best options do not apply in paired-end mode.

A paired-end alignment is reported as a pair of mate alignments, both on a separate line, where the alignment for each mate is formatted the same as an unpaired (singleton) alignment. The alignment for the mate that occurs closest to the beginning of the reference sequence (the "upstream" mate) is always printed before the alignment for the downstream mate. Reads files containing paired-end reads will sometimes name the reads according to whether they are the #1 or #2 mates by appending a /1 or /2 suffix to the read name. If no such suffix is present in Bowtie's input, the suffix will be added when Bowtie prints read names in alignments (except in -S "SAM" mode, where mate information is encoded in the FLAGS field instead).

Finding a valid paired-end alignment where both mates align to repetitive regions of the reference can be very time-consuming. By default, Bowtie avoids much of this cost by imposing a limit on the number of "tries" it makes to match an alignment for one mate with a nearby alignment for the other. The default limit is 100. This causes bowtie to miss some valid paired-end alignments where both mates lie in repetitive regions, but the user may use the --pairtries or -y options to increase Bowtie's sensitivity as desired.

Paired-end alignments where one mate's alignment is entirely contained within the other's are considered invalid.

When colospace alignment is enabled via `-C`, the default setting for paired-end orientation is `--ff`. This is because most SOLiD datasets have that orientation. When colorspace alignment is not enabled (default), the default setting for orientation is `--fr`, since most Illumina datasets have this orientation. The default can be overriden in either case.

Because Bowtie uses an in-memory representation of the original reference string when finding paired-end alignments, its memory footprint is larger when aligning paired-end reads. For example, the human index has a memory footprint of about 2.2 GB in single-end mode and 2.9 GB in paired-end mode. Note that paired-end and unpaired alignment incur the same memory footprint in colorspace (e.g. human incurs about 2.9 GB)

Colorspace Alignment

As of version 0.12.0, `bowtie` can align colorspace reads against a colorspace index when `-C` is specified. Colorspace is the characteristic output format of Applied Biosystems' SOLiD system. In a colorspace read, each character is a color rather than a nucleotide, where a color encodes a class of dinucleotides. E.g. the color blue encodes any of the dinucleotides: AA, CC, GG, TT. Colorspace has the advantage of (often) being able to distinguish sequencing errors from SNPs once the read has been aligned. See ABI's Principles of Di-Base Sequencing document for details.

Colorspace reads

All input formats (FASTA `-f`, FASTQ `-q`, raw `-r`, tab-delimited `--12`, command-line `-c`) are compatible with colorspace (`-C`). When `-C` is specified, read sequences are treated as colors. Colors may be encoded either as numbers (`0`=blue, `1`=green, `2`=orange, `3`=red) or as characters `A/C/G/T` (`A`=blue, `C`=green, `G`=orange, `T`=red).

Some reads include a primer base as the first character; e.g.:

```
>1_53_33_F3
T221312000201030123322122331133 1
>1_53_70_F3
T230211120313123113030011112322 0
...
```

Here, `T` is the primer base. `bowtie` detects and handles primer bases properly (i.e., the primer base and the adjacent color are both trimmed away prior to alignment) as long as the rest of the read is encoded as numbers.

bowtie also handles input in the form of parallel .csfasta and _QV.qual files. Use -f to specify the .csfasta files and -Q (for unpaired reads) or --Q1/--Q2 (for paired-end reads) to specify the corresponding _QV.qual files. It is not necessary to first convert to FASTQ, though bowtie also handles FASTQ-formatted colorspace reads (with -q, the default).

Building a colorspace index

A colorspace index is built in the same way as a normal index except that -C must be specified when running bowtie-build. If the user attempts to use bowtie without -C to align against an index that was built with -C (or vice versa), bowtie prints an error message and quits.

Decoding colorspace alignments

Once a colorspace read is aligned, Bowtie decodes the alignment into nucleotides and reports the decoded nucleotide sequence. A principled decoding scheme is necessary because many different possible decodings are usually possible. Finding the true decoding with 100% certainty requires knowing all variants (e.g. SNPs) in the subject's genome beforehand, which is usually not possible. Instead, bowtie employs the approximate decoding scheme described in the BWA paper. This scheme attempts to distinguish variants from sequencing errors according to their relative likelihood under a model that considers the quality values of the colors and the (configurable) global likelihood of a SNP.

Quality values are also "decoded" so that each reported quality value is a function of the two color qualities overlapping it. Bowtie again adopts the scheme described in the BWA paper, i.e., the decoded nucleotide quality is either the sum of the overlapping color qualities (when both overlapping colors correspond to bases that match in the alignment), the quality of the matching color minus the quality of the mismatching color, or 0 (when both overlapping colors correspond to mismatches).

For accurate decoding, --snpphred/--snpfrac should be set according to the user's best guess of the SNP frequency in the subject. The --snpphred parameter sets the SNP penalty directly (on the Phred quality scale), whereas --snpfrac allows the user to specify the fraction of sites expected to be SNPs; the fraction is then converted to a Phred quality internally. For the purpose of decoding, the SNP fraction is defined in terms of SNPs per *haplotype* base. Thus, if the genome is diploid, heterozygous SNPs have half the weight of homozygous SNPs

Note that in `-S/--sam` mode, the decoded nucleotide sequence is printed for alignments, but the original color sequence (with `A`=blue, `C`=green, `G`=orange, `T`=red) is printed for unaligned reads without any reported alignments. As always, the `--un`, `--max` and `--al` parameters print reads exactly as they appeared in the input file.

Paired-end colorspace alignment

Like other platforms, SOLiD supports generation of paired-end reads. When colorspace alignment is enabled, the default paired-end orientation setting is `--ff`. This is because most SOLiD datasets have that orientation.

Note that SOLiD-generated read files can have "orphaned" mates; i.e. mates without a correpsondingly-named mate in the other file. To avoid problems due to orphaned mates, SOLiD paired-end output should first be converted to `.csfastq` files with unpaired mates omitted. This can be accomplished using, for example, [Galaxy]'s conversion tool (click "NGS: QC and manipulation", then "SOLiD-to-FASTQ" in the left-hand sidebar).

Performance Tuning

1. Use 64-bit bowtie if possible

   The 64-bit version of Bowtie is substantially (usually more then 50%) faster than the 32-bit version, owing to its use of 64-bit arithmetic. If possible, download the 64-bit binaries for Bowtie and run on a 64-bit computer. If you are building Bowtie from sources, you may need to pass the `-m64` option to `g++` to compile the 64-bit version; you can do this by including `BITS=64` in the arguments to the `make` command; e.g.: `make BITS=64 bowtie`. To determine whether your version of bowtie is 64-bit or 32-bit, run `bowtie --version`.

2. If your computer has multiple processors/cores, use `-p`

   The `-p` option causes Bowtie to launch a specified number of parallel search threads. Each thread runs on a different processor/core and all threads find alignments in parallel, increasing alignment throughput by approximately a multiple of the number of threads (though in practice, speedup is somewhat worse than linear).

3. If reporting many alignments per read, try tweaking `bowtie-build --offrate`

   If you are using the `-k`, `-a` or `-m` options and Bowtie is reporting many

alignments per read (an average of more than about 10 per read) and you have some memory to spare, using an index with a denser SA sample can speed things up considerably.

To do this, specify a smaller-than-default `-o/--offrate` value when running `bowtie-build`. A denser SA sample yields a larger index, but is also particularly effective at speeding up alignment when many alignments are reported per read. For example, decreasing the index's `-o/--offrate` by 1 could as much as double alignment performance, and decreasing by 2 could quadruple alignment performance, etc.

On the other hand, decreasing `-o/--offrate` increases the size of the Bowtie index, both on disk and in memory when aligning reads. At the default `-o/--offrate` of 5, the SA sample for the human genome occupies about 375 MB of memory when aligning reads. Decreasing the `-o/--offrate` by 1 doubles the memory taken by the SA sample, and decreasing by 2 quadruples the memory taken, etc.

4. If bowtie "thrashes", try increasing `bowtie --offrate`

   If `bowtie` runs very slow on a relatively low-memory machine (having less than about 4 GB of memory), then try setting `bowtie -o/--offrate` to a *larger* value than the value used to build the index. For example, `bowtie-build`'s default `-o/--offrate` is 5 and all pre-built indexes available from the Bowtie website are built with `-o/--offrate` 5; so if `bowtie` thrashes when querying such an index, try using `bowtie --offrate` 6. If `bowtie` still thrashes, try `bowtie --offrate` 7, etc. A higher `-o/--offrate` causes `bowtie` to use a sparser sample of the suffix array than is stored in the index; this saves memory but makes alignment reporting slower (which is especially slow when using `-a` or large `-k` or `-m`).

Command Line

Usage:

```
bowtie [options]* <ebwt> {-1 <m1> -2 <m2> | --12 <r> | <s>} [<hit>]
```

Main arguments

`<ebwt>` The basename of the index to be searched. The basename is the name of any of the index files up to but not including the final `.1.ebwt` / `.rev.1.ebwt` / etc. `bowtie` looks for the specified index first in the current directory, then in the `indexes` subdirectory under the directory where the `bowtie`

executable is located, then looks in the directory specified in the BOWTIE_INDEXES environment variable.

<m1> Comma-separated list of files containing the #1 mates (filename usually includes _1), or, if -c is specified, the mate sequences themselves. E.g., this might be flyA_1.fq,flyB_1.fq, or, if -c is specified, this might be GGTCATCCT,ACGGGTCGT. Sequences specified with this option must correspond file-for-file and read-for-read with those specified in <m2>. Reads may be a mix of different lengths. If - is specified, bowtie will read the #1 mates from the "standard in" filehandle.

<m2> Comma-separated list of files containing the #2 mates (filename usually includes _2), or, if -c is specified, the mate sequences themselves. E.g., this might be flyA_2.fq,flyB_2.fq, or, if -c is specified, this might be GGTCATCCT,ACGGGTCGT. Sequences specified with this option must correspond file-for-file and read-for-read with those specified in <m1>. Reads may be a mix of different lengths. If - is specified, bowtie will read the #2 mates from the "standard in" filehandle.

<r> Comma-separated list of files containing a mix of unpaired and paired-end reads in Tab-delimited format. Tab-delimited format is a 1-read-per-line format where unpaired reads consist of a read name, sequence and quality string each separated by tabs. A paired-end read consists of a read name, sequnce of the #1 mate, quality values of the #1 mate, sequence of the #2 mate, and quality values of the #2 mate separated by tabs. Quality values can be expressed using any of the scales supported in FASTQ files. Reads may be a mix of different lengths and paired-end and unpaired reads may be intermingled in the same file. If - is specified, bowtie will read the Tab-delimited reads from the "standard in" filehandle.

<s> A comma-separated list of files containing unpaired reads to be aligned, or, if -c is specified, the unpaired read sequences themselves. E.g., this might be lane1.fq,lane2.fq,lane3.fq,lane4.fq, or, if -c is specified, this might be GGTCATCCT,ACGGGTCGT. Reads may be a mix of different lengths. If - is specified, Bowtie gets the reads from the "standard in" filehandle.

Options

File to write alignments to. By default, alignments are written to the "standard out" filehandle (i.e. the console).

Input

`-q`	The query input files (specified either as `<m1>` and `<m2>`, or as `<s>`) are FASTQ files (usually having extension `.fq` or `.fastq`). This is the default. See also: `--solexa-quals` and `--integer-quals`.
`-f`	The query input files (specified either as `<m1>` and `<m2>`, or as `<s>`) are FASTA files (usually having extension `.fa`, `.mfa`, `.fna` or similar). All quality values are assumed to be 40 on the Phred quality scale.
`-r`	The query input files (specified either as `<m1>` and `<m2>`, or as `<s>`) are Raw files: one sequence per line, without quality values or names. All quality values are assumed to be 40 on the Phred quality scale.
`-c`	The query sequences are given on command line. I.e. `<m1>`, `<m2>` and `<singles>` are comma-separated lists of reads rather than lists of read files.
`-C/--color`	Align in colorspace. Read characters are interpreted as colors. The index specified must be a colorspace index (i.e. built with `bowtie-build -C`, or `bowtie` will print an error message and quit. See Colorspace alignment for more details.
`-Q/--quals <files>`	Comma-separated list of files containing quality values for corresponding unpaired CSFASTA reads. Use in combination with `-C` and `-f`. `--integer-quals` is set automatically when `-Q/--quals` is specified.
`--Q1 <files>`	Comma-separated list of files containing quality values for corresponding CSFASTA #1 mates. Use in combination with `-C`, `-f`, and `-1`. `--integer-quals` is set automatically when `--Q1` is specified.

`--Q2 <files>`	Comma-separated list of files containing quality values for corresponding CSFASTA #2 mates. Use in combination with `-C`, `-f`, and `-2`. `--integer-quals` is set automatically when `--Q2` is specified.
`-s/--skip <int>`	Skip (i.e. do not align) the first `<int>` reads or pairs in the input.
`-u/--qupto <int>`	Only align the first `<int>` reads or read pairs from the input (after the `-s/--skip` reads or pairs have been skipped). Default: no limit.
`-5/--trim5 <int>`	Trim `<int>` bases from high-quality (left) end of each read before alignment (default: 0).
`-3/--trim3 <int>`	Trim `<int>` bases from low-quality (right) end of each read before alignment (default: 0).
`--phred33-quals`	Input qualities are ASCII chars equal to the Phred quality plus 33. Default: on.
`--phred64-quals`	Input qualities are ASCII chars equal to the Phred quality plus 64. Default: off.
`--solexa-quals`	Convert input qualities from Solexa (which can be negative) to Phred (which can't). This is usually the right option for use with (unconverted) reads emitted by GA Pipeline versions prior to 1.3. Default: off.
`--solexa1.3-quals`	Same as `--phred64-quals`. This is usually the right option for use with (unconverted) reads emitted by GA Pipeline version 1.3 or later. Default: off.
`--integer-quals`	Quality values are represented in the read input file as space-separated ASCII integers, e.g., `40 40 30 40`..., rather than ASCII

characters, e.g., II?I.... Integers are treated as being on the Phred quality scale unless --solexa-quals is also specified. Default: off.

Alignment

`-v <int>`

Report alignments with at most <int> mismatches. -e and -l options are ignored and quality values have no effect on what alignments are valid. -v is mutually exclusive with -n.

`-n/--seedmms <int>`

Maximum number of mismatches permitted in the "seed", i.e. the first L base pairs of the read (where L is set with -l/--seedlen). This may be 0, 1, 2 or 3 and the default is 2. This option is mutually exclusive with the -v option.

`-e/--maqerr <int>`

Maximum permitted total of quality values at *all* mismatched read positions throughout the entire alignment, not just in the "seed". The default is 70. Like Maq, bowtie rounds quality values to the nearest 10 and saturates at 30; rounding can be disabled with --nomaqround.

`-l/--seedlen <int>`

The "seed length"; i.e., the number of bases on the high-quality end of the read to which the -n ceiling applies. The lowest permitted setting is 5 and the default is 28. bowtie is faster for larger values of -l.

`--nomaqround`

Maq accepts quality values in the Phred quality scale, but internally rounds values to the nearest 10, with a maximum of 30. By default, bowtie also rounds this way. --nomaqround prevents this rounding in bowtie.

`-I/--minins <int>`

The minimum insert size for valid paired-end alignments. E.g. if -I 60 is specified and a paired-end alignment consists of two 20-bp alignments in the appropriate orientation with a 20-bp gap between them, that alignment is considered valid (as long as -X is also satisfied). A 19-bp gap would not be valid in that case. If trimming options -3 or -5 are also used, the -I

constraint is applied with respect to the untrimmed mates. Default: 0.

`-X/--maxins <int>`: The maximum insert size for valid paired-end alignments. E.g. if `-X 100` is specified and a paired-end alignment consists of two 20-bp alignments in the proper orientation with a 60-bp gap between them, that alignment is considered valid (as long as `-I` is also satisfied). A 61-bp gap would not be valid in that case. If trimming options `-3` or `-5` are also used, the `-X` constraint is applied with respect to the untrimmed mates, not the trimmed mates. Default: 250.

`--fr/--rf/--ff`: The upstream/downstream mate orientations for a valid paired-end alignment against the forward reference strand. E.g., if `--fr` is specified and there is a candidate paired-end alignment where mate1 appears upstream of the reverse complement of mate2 and the insert length constraints are met, that alignment is valid. Also, if mate2 appears upstream of the reverse complement of mate1 and all other constraints are met, that too is valid. `--rf` likewise requires that an upstream mate1 be reverse-complemented and a downstream mate2 be forward-oriented. `--ff` requires both an upstream mate1 and a downstream mate2 to be forward-oriented. Default: `--fr` when `-C` (colorspace alignment) is not specified, `--ff` when `-C` is specified.

`--nofw/--norc`: If `--nofw` is specified, `bowtie` will not attempt to align against the forward reference strand. If `--norc` is specified, `bowtie` will not attempt to align against the reverse-complement reference strand. For paired-end reads using `--fr` or `--rf` modes, `--nofw` and `--norc` apply to the forward and reverse-complement pair orientations. I.e. specifying `--nofw` and `--fr` will only find reads in the R/F orientation where mate 2 occurs upstream of mate 1 with respect to the forward reference strand.

`--maxbts`: The maximum number of backtracks permitted when aligning a read in `-n` 2 or `-n` 3 mode (default: 125 without `--best`, 800 with `--best`). A "backtrack" is the introduction of a speculative substitution into the alignment. Without this limit, the default parameters will sometimes require that `bowtie` try 100s or

1,000s of backtracks to align a read, especially if the read has many low-quality bases and/or has no valid alignments, slowing bowtie down significantly. However, this limit may cause some valid alignments to be missed. Higher limits yield greater sensitivity at the expensive of longer running times. See also: -y/--tryhard.

--pairtries <int>

For paired-end alignment, this is the maximum number of attempts bowtie will make to match an alignment for one mate up with an alignment for the opposite mate. Most paired-end alignments require only a few such attempts, but pairs where both mates occur in highly repetitive regions of the reference can require significantly more. Setting this to a higher number allows bowtie to find more paired- end alignments for repetitive pairs at the expense of speed. The default is 100. See also: -y/--tryhard.

-y/--tryhard

Try as hard as possible to find valid alignments when they exist, including paired-end alignments. This is equivalent to specifying very high values for the --maxbts and --pairtries options. This mode is generally much slower than the default settings, but can be useful for certain problems. This mode is slower when (a) the reference is very repetitive, (b) the reads are low quality, or (c) not many reads have valid alignments.

--chunkmbs <int>

The number of megabytes of memory a given thread is given to store path descriptors in --best mode. Best-first search must keep track of many paths at once to ensure it is always extending the path with the lowest cumulative cost. Bowtie tries to minimize the memory impact of the descriptors, but they can still grow very large in some cases. If you receive an error message saying that chunk memory has been exhausted in --best mode, try adjusting this parameter up to dedicate more memory to the descriptors. Default: 64.

Reporting

-k <int> Report up to <int> valid alignments per read or pair (default: 1). Validity of alignments is determined by the alignment policy

(combined effects of -n, -v, -l, and -e). If more than one valid alignment exists and the --best and --strata options are specified, then only those alignments belonging to the best alignment "stratum" will be reported. Bowtie is designed to be very fast for small -k but bowtie can become significantly slower as -k increases. If you would like to use Bowtie for larger values of -k, consider building an index with a denser suffix-array sample, i.e. specify a smaller -o/--offrate when invoking bowtie-build for the relevant index (see the Performance tuning section for details).

-a/--all

Report all valid alignments per read or pair (default: off). Validity of alignments is determined by the alignment policy (combined effects of -n, -v, -l, and -e). If more than one valid alignment exists and the --best and --strata options are specified, then only those alignments belonging to the best alignment "stratum" will be reported. Bowtie is designed to be very fast for small -k but bowtie can become significantly slower if -a/--all is specified. If you would like to use Bowtie with -a, consider building an index with a denser suffix-array sample, i.e. specify a smaller -o/--offrate when invoking bowtie-build for the relevant index (see the Performance tuning section for details).

-m <int>

Suppress all alignments for a particular read or pair if more than <int> reportable alignments exist for it. Reportable alignments are those that would be reported given the -n, -v, -l, -e, -k, -a, --best, and --strata options. Default: no limit. Bowtie is designed to be very fast for small -m but bowtie can become significantly slower for larger values of -m. If you would like to use Bowtie for larger values of -k, consider building an index with a denser suffix-array sample, i.e. specify a smaller -o/--offrate when invoking bowtie-build for the relevant index (see the Performance tuning section for details).

-M <int>

Behaves like -m except that if a read has more than <int> reportable alignments, one is reported at random. In default output mode, the selected alignment's 7th column is set to <int>+1 to indicate the read has at least <int>+1 valid alignments. In -S/--sam mode, the selected alignment is given a MAPQ (mapping quality) of 0 and the XM:I field is set to <int>+1. This option requires --best; if specified without --best, --best is enabled automatically.

`--best` Make Bowtie guarantee that reported singleton alignments are "best" in terms of stratum (i.e. number of mismatches, or mismatches in the seed in the case of `-n` mode) and in terms of the quality values at the mismatched position(s). Stratum always trumps quality; e.g. a 1-mismatch alignment where the mismatched position has Phred quality 40 is preferred over a 2-mismatch alignment where the mismatched positions both have Phred quality 10. When `--best` is not specified, Bowtie may report alignments that are sub-optimal in terms of stratum and/or quality (though an effort is made to report the best alignment). `--best` mode also removes all strand bias. Note that `--best` does not affect which alignments are considered "valid" by `bowtie`, only which valid alignments are reported by `bowtie`. When `--best` is specified and multiple hits are allowed (via `-k` or `-a`), the alignments for a given read are guaranteed to appear in best-to-worst order in `bowtie`'s output. `bowtie` is somewhat slower when `--best` is specified.

`--strata` If many valid alignments exist and are reportable (e.g. are not disallowed via the `-k` option) and they fall into more than one alignment "stratum", report only those alignments that fall into the best stratum. By default, Bowtie reports all reportable alignments regardless of whether they fall into multiple strata. When `--strata` is specified, `--best` must also be specified.

Output

`-t/--time` Print the amount of wall-clock time taken by each phase.

`-B/--offbase <int>` When outputting alignments, number the first base of a reference sequence as `<int>`. Default: 0.

`--quiet` Print nothing besides alignments.

`--refout` Write alignments to a set of files named `refXXXXX.map`, where `XXXXX` is the 0-padded index of the reference sequence aligned to. This can be a useful way to break up work for downstream analyses when dealing with, for example, large numbers of reads aligned to the assembled human genome. If `<hits>` is also

specified, it will be ignored.

`--refidx` When a reference sequence is referred to in a reported alignment, refer to it by 0-based index (its offset into the list of references that were indexed) rather than by name.

`--al <filename>` Write all reads for which at least one alignment was reported to a file with name `<filename>`. Written reads will appear as they did in the input, without any of the trimming or translation of quality values that may have taken place within `bowtie`. Paired-end reads will be written to two parallel files with `_1` and `_2` inserted in the filename, e.g., if `<filename>` is `aligned.fq`, the #1 and #2 mates that fail to align will be written to `aligned_1.fq` and `aligned_2.fq` respectively.

`--un <filename>` Write all reads that could not be aligned to a file with name `<filename>`. Written reads will appear as they did in the input, without any of the trimming or translation of quality values that may have taken place within Bowtie. Paired-end reads will be written to two parallel files with `_1` and `_2` inserted in the filename, e.g., if `<filename>` is `unaligned.fq`, the #1 and #2 mates that fail to align will be written to `unaligned_1.fq` and `unaligned_2.fq` respectively. Unless `--max` is also specified, reads with a number of valid alignments exceeding the limit set with the `-m` option are also written to `<filename>`.

`--max <filename>` Write all reads with a number of valid alignments exceeding the limit set with the `-m` option to a file with name `<filename>`. Written reads will appear as they did in the input, without any of the trimming or translation of quality values that may have taken place within `bowtie`. Paired-end reads will be written to two parallel files with `_1` and `_2` inserted in the filename, e.g., if `<filename>` is `max.fq`, the #1 and #2 mates that exceed the `-m` limit will be written to `max_1.fq` and `max_2.fq` respectively. These reads are not written to the file specified with `--un`.

`--suppress <cols>` Suppress columns of output in the default output mode. E.g. if `--suppress 1,5,6` is specified, the read name, read sequence, and read quality fields will be omitted. See Default Bowtie output

for field descriptions. This option is ignored if the output mode is `-S/--sam`.

`--fullref` Print the full refernce sequence name, including whitespace, in alignment output. By default `bowtie` prints everything up to but not including the first whitespace.

Colorspace

`--snpphred <int>` When decoding colorspace alignments, use `<int>` as the SNP penalty. This should be set to the user's best guess of the true ratio of SNPs per base in the subject genome, converted to the Phred quality scale. E.g., if the user expects about 1 SNP every 1,000 positions, `--snpphred` should be set to 30 (which is also the default). To specify the fraction directly, use `--snpfrac`.

`--snpfrac <dec>` When decoding colorspace alignments, use `<dec>` as the estimated ratio of SNPs per base. For best decoding results, this should be set to the user's best guess of the true ratio. `bowtie` internally converts the ratio to a Phred quality, and behaves as if that quality had been set via the `--snpphred` option. Default: 0.001.

`--col-cseq` If reads are in colorspace and the default output mode is active, `--col-cseq` causes the reads' color sequence to appear in the read-sequence column (column 5) instead of the decoded nucleotide sequence. See the Decoding colorspace alignments section for details about decoding. This option is ignored in `-S/--sam` mode.

`--col-cqual` If reads are in colorspace and the default output mode is active, `--col-cqual` causes the reads' original (color) quality sequence to appear in the quality column (column 6) instead of the decoded qualities. See the Colorspace alignment section for details about decoding. This option is ignored in `-S/--sam` mode.

`--col-keepends`	When decoding colorpsace alignments, `bowtie` trims off a nucleotide and quality from the left and right edges of the alignment. This is because those nucleotides are supported by only one color, in contrast to the middle nucleotides which are supported by two. Specify `--col-keepends` to keep the extreme-end nucleotides and qualities.

SAM

`-S/--sam`	Print alignments in SAM format. See the SAM output section of the manual for details. To suppress all SAM headers, use `--sam-nohead` in addition to `-S/--sam`. To suppress just the `@SQ` headers (e.g. if the alignment is against a very large number of reference sequences), use `--sam-nosq` in addition to `-S/--sam`. `bowtie` does not write BAM files directly, but SAM output can be converted to BAM on the fly by piping `bowtie`'s output to `samtools view`. `-S/--sam` is not compatible with `--refout`.
`--mapq <int>`	If an alignment is non-repetitive (according to `-m`, `--strata` and other options) set the `MAPQ` (mapping quality) field to this value. See the SAM Spec for details about the `MAPQ` field Default: 255.
`--sam-nohead`	Suppress header lines (starting with `@`) when output is `-S/--sam`. This must be specified *in addition to* `-S/--sam`. `--sam-nohead` is ignored unless `-S/--sam` is also specified.
`--sam-nosq`	Suppress `@SQ` header lines when output is `-S/--sam`. This must be specified *in addition to* `-S/--sam`. `--sam-nosq` is ignored unless `-S/--sam` is also specified.
`--sam-RG <text>`	Add `<text>` (usually of the form `TAG:VAL`, e.g. `ID:IL7LANE2`) as a field on the `@RG` header line. Specify `--sam-RG` multiple times to set multiple fields. See the SAM Spec for details about what fields are legal. Note that if any `@RG` fields are set using this option, the `ID` and `SM` fields must both be among them to make the `@RG` line legal according to the SAM Spec. `--sam-RG` is ignored unless `-S/--sam` is also specified.

Performance

`-o/--offrate <int>`	Override the offrate of the index with `<int>`. If `<int>` is greater than the offrate used to build the index, then some row markings are discarded when the index is read into memory. This reduces the memory footprint of the aligner but requires

more time to calculate text offsets. <int> must be greater than the value used to build the index.

-p/--threads <int> Launch <int> parallel search threads (default: 1). Threads will run on separate processors/cores and synchronize when parsing reads and outputting alignments. Searching for alignments is highly parallel, and speedup is fairly close to linear. This option is only available if bowtie is linked with the pthreads library (i.e. if BOWTIE_PTHREADS=0 is not specified at build time).

--mm Use memory-mapped I/O to load the index, rather than normal C file I/O. Memory-mapping the index allows many concurrent bowtie processes on the same computer to share the same memory image of the index (i.e. you pay the memory overhead just once). This facilitates memory-efficient parallelization of bowtie in situations where using -p is not possible.

--shmem Use shared memory to load the index, rather than normal C file I/O. Using shared memory allows many concurrent bowtie processes on the same computer to share the same memory image of the index (i.e. you pay the memory overhead just once). This facilitates memory-efficient parallelization of bowtie in situations where using -p is not desirable. Unlike --mm, --shmem installs the index into shared memory permanently, or until the user deletes the shared memory chunks manually. See your operating system documentation for details on how to manually list and remove shared memory chunks (on Linux and Mac OS X, these commands are ipcs and ipcrm). You may also need to increase your OS's maximum shared-memory chunk size to accomodate larger indexes; see your OS documentation.

Other

--seed <int> Use <int> as the seed for pseudo-random number generator.

`--verbose` Print verbose output (for debugging).

`--version` Print version information and quit.

`-h/--help` Print usage information and quit.

Default `bowtie` output

`bowtie` outputs one alignment per line. Each line is a collection of 8 fields separated by tabs; from left to right, the fields are:

1. Name of read that aligned
2. Reference strand aligned to, `+` for forward strand, `-` for reverse
3. Name of reference sequence where alignment occurs, or numeric ID if no name was provided
4. 0-based offset into the forward reference strand where leftmost character of the alignment occurs
5. Read sequence (reverse-complemented if orientation is `-`).

   If the read was in colorspace, then the sequence shown in this column is the sequence of *decoded nucleotides*, not the original colors. See the Colorspace alignment section for details about decoding. To display colors instead, use the `--col-cseq` option.
6. ASCII-encoded read qualities (reversed if orientation is `-`). The encoded quality values are on the Phred scale and the encoding is ASCII-offset by 33 (ASCII char `!`).

   If the read was in colorspace, then the qualities shown in this column are the *decoded qualities*, not the original qualities. See the Colorspace alignment section for details about decoding. To display colors instead, use the `--col-cqual` option.
7. If `-M` was specified and the prescribed ceiling was exceeded for this read, this column contains the value of the ceiling, indicating that at least that many valid alignments were found in addition to the one reported.

Otherwise, this column contains the number of other instances where the same sequence aligned against the same reference characters as were aligned against in the reported alignment. This is *not* the number of other places the read aligns with the same number of mismatches. The number in this column is generally not a good proxy for that number (e.g., the number in this column may be '0' while the number of other alignments with the same number of mismatches might be large).

8. Comma-separated list of mismatch descriptors. If there are no mismatches in the alignment, this field is empty. A single descriptor has the format offset:reference-base>read-base. The offset is expressed as a 0-based offset from the high-quality (5') end of the read.

SAM `bowtie` output

Following is a brief description of the SAM format as output by `bowtie` when the `-S/--sam` option is specified. For more details, see the SAM format specification.

When `-S/--sam` is specified, `bowtie` prints a SAM header with `@HD`, `@SQ` and `@PG` lines. When one or more `--sam-RG` arguments are specified, `bowtie` will also print an `@RG` line that includes all user-specified `--sam-RG` tokens separated by tabs.

Each subsequnt line corresponds to a read or an alignment. Each line is a collection of at least 12 fields separated by tabs; from left to right, the fields are:

1. Name of read that aligned
2. Sum of all applicable flags. Flags relevant to Bowtie are:

   1 The read is one of a pair

   2 The alignment is one end of a proper paired-end alignment

   4 The read has no reported alignments

   8 The read is one of a pair and has no reported alignments

   16 The alignment is to the reverse reference strand

32 The other mate in the paired-end alignment is aligned to the reverse reference strand

64 The read is the first (#1) mate in a pair

128 The read is the second (#2) mate in a pair

Thus, an unpaired read that aligns to the reverse reference strand will have flag 16. A paired-end read that aligns and is the first mate in the pair will have flag 83 (= 64 + 16 + 2 + 1).

3. Name of reference sequence where alignment occurs, or ordinal ID if no name was provided
4. 1-based offset into the forward reference strand where leftmost character of the alignment occurs
5. Mapping quality
6. CIGAR string representation of alignment
7. Name of reference sequence where mate's alignment occurs. Set to `=` if the mate's reference sequence is the same as this alignment's, or `*` if there is no mate.
8. 1-based offset into the forward reference strand where leftmost character of the mate's alignment occurs. Offset is 0 if there is no mate.
9. Inferred insert size. Size is negative if the mate's alignment occurs upstream of this alignment. Size is 0 if there is no mate.
10. Read sequence (reverse-complemented if aligned to the reverse strand)
11. ASCII-encoded read qualities (reverse-complemented if the read aligned to the reverse strand). The encoded quality values are on the Phred quality scale and the encoding is ASCII-offset by 33 (ASCII char `!`), similarly to a FASTQ file.
12. Optional fields. Fields are tab-separated. For descriptions of all possible optional fields, see the SAM format specification. `bowtie` outputs some of these optional fields for each alignment, depending on the type of the alignment:

`NM:i:<N>` Aligned read has an edit distance of `<N>`.

`CM:i:<N>` Aligned read has an edit distance of `<N>` in colorspace. This field is present in addition to the `NM` field in `-C/--color` mode, but is omitted otherwise.

`MD:Z:<S>` For aligned reads, `<S>` is a string representation of the mismatched reference bases in the alignment. See SAM format specification for details. For colorspace alignments, `<S>` describes the decoded *nucleotide* alignment, not the colorspace alignment.

`XA:i:<N>` Aligned read belongs to stratum `<N>`. See Strata for definition.

`XM:i:<N>` For a read with no reported alignments, `<N>` is 0 if the read had no alignments. If `-m` was specified and the read's alignments were supressed because the `-m` ceiling was exceeded, `<N>` equals the `-m` ceiling + 1, to indicate that there were at least that many valid alignments (but all were suppressed). In `-M` mode, if the alignment was randomly selected because the `-M` ceiling was exceeded, `<N>` equals the `-M` ceiling + 1, to indicate that there were at least that many valid alignments (of which one was reported at random).

The `bowtie-build` indexer

`bowtie-build` builds a Bowtie index from a set of DNA sequences. `bowtie-build` outputs a set of 6 files with suffixes `.1.ebwt`, `.2.ebwt`, `.3.ebwt`, `.4.ebwt`, `.rev.1.ebwt`, and `.rev.2.ebwt`. These files together constitute the index: they are all that is needed to align reads to that reference. The original sequence files are no longer used by Bowtie once the index is built.

Use of Karkkainen's [blockwise algorithm] allows `bowtie-build` to trade off between running time and memory usage. `bowtie-build` has three options governing how it makes this trade: `-p/--packed`, `--bmax`/`--bmaxdivn`, and `--dcv`. By default, `bowtie-build` will automatically search for the settings that yield the best running time without exhausting memory. This behavior can be disabled using the `-a/--noauto` option.

The indexer provides options pertaining to the "shape" of the index, e.g.

`--offrate` governs the fraction of Burrows-Wheeler rows that are "marked" (i.e., the density of the suffix-array sample; see the original FM Index paper for details). All of these options are potentially profitable trade-offs depending on the application. They have been set to defaults that are reasonable for most cases according to our experiments. See [Performance Tuning] for details.

Because `bowtie-build` uses 32-bit pointers internally, it can handle up to a theoretical maximum of 2^32-1 (somewhat more than 4 billion) characters in an index, though, with other constraints, the actual ceiling is somewhat less than that. If your reference exceeds 2^32-1 characters, `bowtie-build` will print an error message and abort. To resolve this, divide your reference sequences into smaller batches and/or chunks and build a separate index for each.

If your computer has more than 3-4 GB of memory and you would like to exploit that fact to make index building faster, use a 64-bit version of the `bowtie-build` binary. The 32-bit version of the binary is restricted to using less than 4 GB of memory. If a 64-bit pre-built binary does not yet exist for your platform on the sourceforge download site, you will need to build one from source.

The Bowtie index is based on the FM Index of Ferragina and Manzini, which in turn is based on the Burrows-Wheeler transform. The algorithm used to build the index is based on the [blockwise algorithm] of Karkkainen.

Command Line

Usage:

```
bowtie-build [options]* <reference_in> <ebwt_base>
```

Main arguments

`<reference_in>`	A comma-separated list of FASTA files containing the reference sequences to be aligned to, or, if `-c` is specified, the sequences themselves. E.g., `<reference_in>` might be `chr1.fa,chr2.fa,chrX.fa,chrY.fa`, or, if `-c` is specified, this might be `GGTCATCCT,ACGGGTCGT,CCGTTCTATGCGGCTTA`.
`<ebwt_base>`	The basename of the index files to write. By default, `bowtie-build` writes files named `NAME.1.ebwt`, `NAME.2.ebwt`, `NAME.3.ebwt`, `NAME.4.ebwt`, `NAME.rev.1.ebwt`, and `NAME.rev.2.ebwt`, where `NAME` is `<ebwt_base>`.

Options

`-f`	The reference input files (specified as `<reference_in>`) are FASTA files (usually having extension `.fa`, `.mfa`, `.fna` or similar).
`-c`	The reference sequences are given on the command line. I.e. `<reference_in>` is a comma-separated list of sequences rather than a list of FASTA files.
`-C/--color`	Build a colorspace index, to be queried using `bowtie -C`.
`-a/--noauto`	Disable the default behavior whereby `bowtie-build` automatically selects values for the `--bmax`, `--dcv` and `--packed` parameters according to available memory. Instead, user may specify values for those parameters. If memory is exhausted during indexing, an error message will be printed; it is up to the user to try new parameters.
`-p/--packed`	Use a packed (2-bits-per-nucleotide) representation for DNA strings. This saves memory but makes indexing 2-3 times slower. Default: off. This is configured automatically by default; use `-a/--noauto` to configure manually.
`--bmax <int>`	The maximum number of suffixes allowed in a block. Allowing more suffixes per block makes indexing faster, but increases peak memory usage. Setting this option overrides any previous setting for `--bmax`, or `--bmaxdivn`. Default (in terms of the `--bmaxdivn` parameter) is `--bmaxdivn` 4. This is configured automatically by default; use `-a/--noauto` to configure manually.
`--bmaxdivn <int>`	The maximum number of suffixes allowed in a block, expressed as a fraction of the length of the reference. Setting this option overrides any previous setting for `--bmax`, or `--bmaxdivn`. Default: `--bmaxdivn` 4. This is configured automatically by default; use `-a/--noauto` to configure manually.

`--dcv <int>`	Use `<int>` as the period for the difference-cover sample. A larger period yields less memory overhead, but may make suffix sorting slower, especially if repeats are present. Must be a power of 2 no greater than 4096. Default: 1024. This is configured automatically by default; use `-a/--noauto` to configure manually.
`--nodc`	Disable use of the difference-cover sample. Suffix sorting becomes quadratic-time in the worst case (where the worst case is an extremely repetitive reference). Default: off.
`-r/--noref`	Do not build the `NAME.3.ebwt` and `NAME.4.ebwt` portions of the index, which contain a bitpacked version of the reference sequences and are used for paired-end alignment.
`-3/--justref`	Build *only* the `NAME.3.ebwt` and `NAME.4.ebwt` portions of the index, which contain a bitpacked version of the reference sequences and are used for paired-end alignment.
`-o/--offrate <int>`	To map alignments back to positions on the reference sequences, it's necessary to annotate ("mark") some or all of the Burrows-Wheeler rows with their corresponding location on the genome. `-o/--offrate` governs how many rows get marked: the indexer will mark every 2^`<int>` rows. Marking more rows makes reference-position lookups faster, but requires more memory to hold the annotations at runtime. The default is 5 (every 32nd row is marked; for human genome, annotations occupy about 340 megabytes).
`-t/--ftabchars <int>`	The ftab is the lookup table used to calculate an initial Burrows-Wheeler range with respect to the first `<int>` characters of the query. A larger `<int>` yields a larger lookup table but faster query times. The ftab has size 4^(`<int>`+1) bytes. The default setting is 10 (ftab is 4MB).
`--ntoa`	Convert Ns in the reference sequence to As before building the index. By default, Ns are simply excluded from the index

	and `bowtie` will not report alignments that overlap them.
`--big --little`	Endianness to use when serializing integers to the index file. Default: little-endian (recommended for Intel- and AMD-based architectures).
`--seed <int>`	Use `<int>` as the seed for pseudo-random number generator.
`--cutoff <int>`	Index only the first `<int>` bases of the reference sequences (cumulative across sequences) and ignore the rest.
`-q/--quiet`	`bowtie-build` is verbose by default. With this option `bowtie-build` will print only error messages.
`-h/--help`	Print usage information and quit.
`--version`	Print version information and quit.

The `bowtie-inspect` index inspector

`bowtie-inspect` extracts information from a Bowtie index about what kind of index it is and what reference sequences were used to build it. When run without any options, the tool will output a FASTA file containing the sequences of the original references (with all non-`A`/`C`/`G`/`T` characters converted to `N`s). It can also be used to extract just the reference sequence names using the `-n/--names` option or a more verbose summary using the `-s/--summary` option.

Command Line

Usage:

```
bowtie-inspect [options]* <ebwt_base>
```

Main arguments

`<ebwt_base>` The basename of the index to be inspected. The basename is name of any of the index files but with the `.X.ebwt` or `.rev.X.ebwt` suffix omitted. `bowtie-inspect` first looks in the current directory for the index files, then looks in the `indexes` subdirectory under the directory where the currently-running `bowtie` executable is located, then looks in the directory specified in the `BOWTIE_INDEXES` environment variable.

Options

`-a/--across <int>` When printing FASTA output, output a newline character every `<int>` bases (default: 60).

`-n/--names` Print reference sequence names, one per line, and quit.

`-s/--summary` Print a summary that includes information about index settings, as well as the names and lengths of the input sequences. The summary has this format:

```
Colorspace  <0 or 1>
SA-Sample   1 in <sample>
FTab-Chars  <chars>
Sequence-1  <name>  <len>
Sequence-2  <name>  <len>
...
Sequence-N  <name>  <len>
```

Fields are separated by tabs.

`-e/--ebwt-ref` By default, when `bowtie-inspect` is run without `-s` or `-n`, it recreates the reference nucleotide sequences using the bit-encoded reference nucleotides kept in the `.3.ebwt` and `.4.ebwt` index files. When `-e/--ebwt-ref` is specified, `bowtie-inspect` recreates the reference sequences from the Burrows-Wheeler-transformed reference sequence in the `.1.ebwt` file instead. The reference recreation process is much slower when `-e/--ebwt-ref` is specified. Also, when `-e/--ebwt-ref` is specified and the index is in colorspace, the reference is printed in colors (A=blue, C=green, G=orange, T=red).

`-v/--verbose` Print verbose output (for debugging).

`--version` Print version information and quit.

`-h/--help` Print usage information and quit.

This research was supported in part by NIH grants R01-LM006845, R01-GM083873 and P41HG004059.

Administrator: Ben Langmead.
Design by David Herreman