Resequencing and Mapping

Andreas Gisel
Inernational Institute of Tropical Agriculture (IITA)
Ibadan, Nigeria

The Principle of Mapping

reads

```
good, ood_, d_mo, morn, orni, ning, ing_,
g_be, beau, auti, utif, iful, ul_w, _wor orld
```

reference

```
good_morning_beautiful_world
```

mapping

```
          ing_    utif
   d_mo ning    auti    _wor
 ood_ orni    beau    ul_w
good morn   g_be    iful   orld

good_morning_beautiful_world
```

consensus

```
good_morning_beautiful_world
```

What can we find with Mapping

reads

```
good, ood_, d_ev, even, veni, ning, ing_,
g_be, beau, auti, utif, iful, ul_w, _wor orld
```

reference

```
good_morning_beautiful_world
```

mapping

```
          ing_    utif
   d_ev ning    auti    _wor
 ood_ veni    beau    ul_w
good even  g_be    iful  orld

good_morning_beautiful_world
```

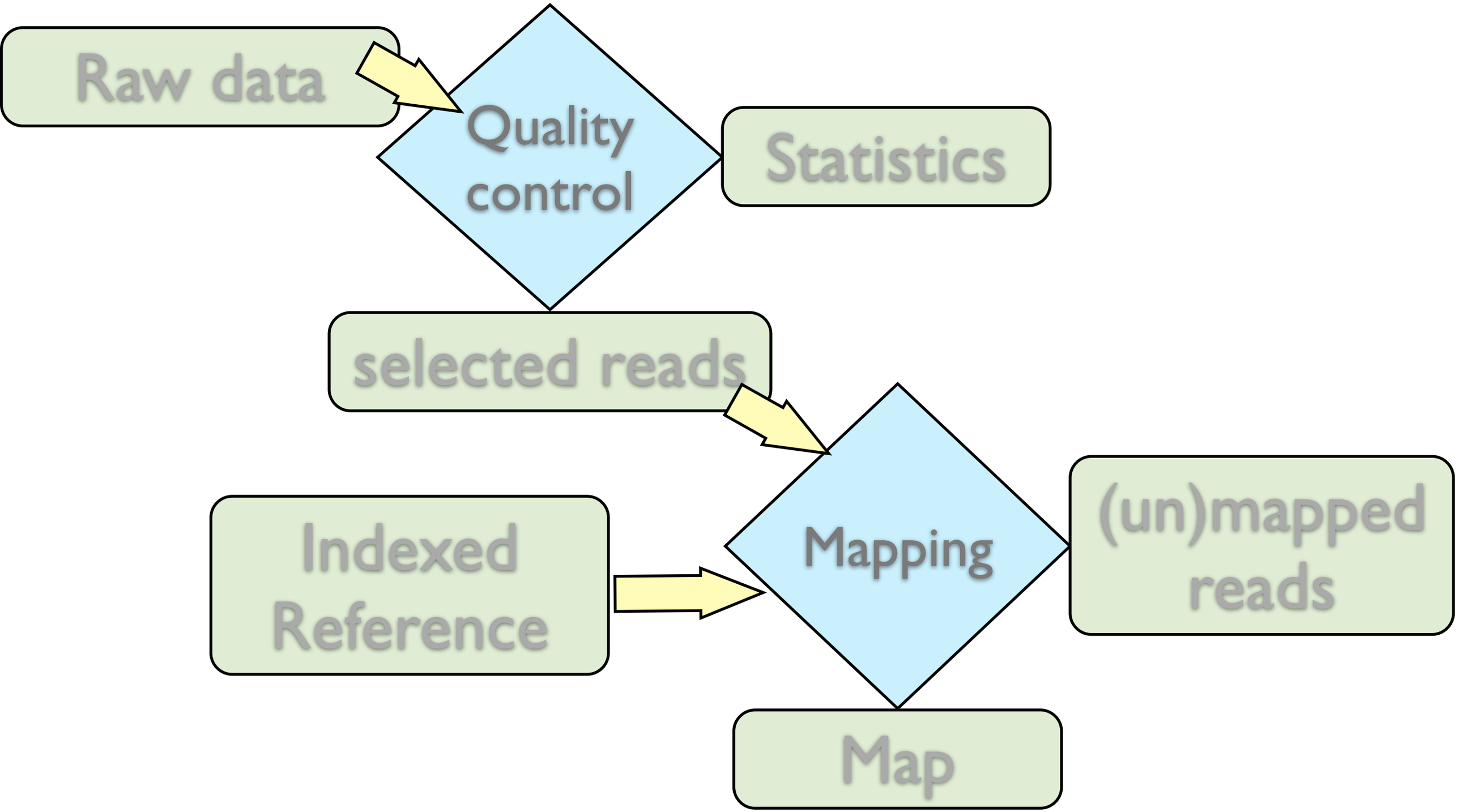
consensus

```
good_evening_beautiful_world
```

What can we find with Mapping

reference	`good_morning_beautiful_world`	
consensus	`good_evening_beautiful_world`	**change**
consensus	`mood_morning_beautiful_world`	**SNP**
consensus	`good_morning _world`	**deletion**
consensus	`good_morning beautiful_world` + not mappable reads	**insert**
consensus	`good_evening dlrow_lufituaeb` + not mappable reads	**inversion**

Workflow for Mapping



Compare Mapping Tools

Table 1. Evaluation on simulated data

	Single-end			Paired-end		
Program	Time (s)	Conf (%)	Err (%)	Time (s)	Conf (%)	Err (%)
Bowtie-32	1271	79.0	0.76	1391	85.7	0.57
BWA-32	823	80.6	0.30	1224	89.6	0.32
MAQ-32	19797	81.0	0.14	21589	87.2	0.07
SOAP2-32	256	78.6	1.16	1909	86.8	0.78
Bowtie-70	1726	86.3	0.20	1580	90.7	0.43
BWA-70	1599	90.7	0.12	1619	96.2	0.11
MAQ-70	17928	91.0	0.13	19046	94.6	0.05
SOAP2-70	317	90.3	0.39	708	94.5	0.34
bowtie-125	1966	88.0	0.07	1701	91.0	0.37
BWA-125	3021	93.0	0.05	3059	97.6	0.04
MAQ-125	17506	92.7	0.08	19388	96.3	0.02
SOAP2-125	555	91.5	0.17	1187	90.8	0.14

One million pairs of 32, 70 and 125 bp reads, respectively, were simulated from the human genome with 0.09% SNP mutation rate, 0.01% indel mutation rate and 2% uniform sequencing base error rate. The insert size of 32 bp reads is drawn from a normal distribution $N(170,25)$, and of 70 and 125 bp reads from $N(500,50)$. CPU time in seconds on a single core of a 2.5 GHz Xeon E5420 processor (Time), percent confidently mapped reads (Conf) and percent erroneous alignments out of confident mappings (Err) are shown in the table.

Compare Mapping Tools

Table 1: Popular short-read alignment software

Program	Algorithm	SOLiD	Long[a]	Gapped	PE[b]	Q[c]
Bfast	hashing ref.	Yes	No	Yes	Yes	No
Bowtie	FM-index	Yes	No	No	Yes	Yes
BWA	FM-index	Yes[d]	Yes[e]	Yes	Yes	No
MAQ	hashing reads	Yes	No	Yes[f]	Yes	Yes
Mosaik	hashing ref.	Yes	Yes	Yes	Yes	No
Novoalign[g]	hashing ref.	No	No	Yes	Yes	Yes

[a]Work well for Sanger and 454 reads, allowing gaps and clipping. [b]Paired end mapping. [c]Make use of base quality in alignment. [d]BWA trims the primer base and the first color for a color read. [e]Long-read alignment implemented in the BWA-SW module. [f]MAQ only does gapped alignment for Illumina paired-end reads. [g]Free executable for non-profit projects only.

Tools for Mapping

bowtie - an ultrafast, memory-efficient short read aligner
http://bowtie-bio.sourceforge.net/index.shtml

BWA - Burrows-Wheeler Aligner
http://bio-bwa.sourceforge.net/

SOAPaligner - Short Oligonucleotide Analysis Package
http://soap.genomics.org.cn/soapaligner.html

bowtie

Bowtie

An ultrafast memory-efficient short read aligner

JOHNS HOPKINS UNIVERSITY

Bowtie is an ultrafast, memory-efficient short read aligner. It aligns short DNA sequences (reads) to the human genome at a rate of over 25 million 35-bp reads per hour. Bowtie indexes the genome with a Burrows-Wheeler index to keep its memory footprint small: typically about 2.2 GB for the human genome (2.9 GB for paired-end).

OSI certified

Recent news

Hiring Postdocs

- The Langmead and Salzberg labs have open positions for postdoctoral researchers. See the posting and please apply if you're interested in working with either or both of us.

0.12.9 release - 12/16/12

- Fixed a bug whereby read names would not be truncated at first whitespace character in unmapped or maxed-out SAM records.
- Fixed errors and warnings when compiling with `clang++`.
- Fixed most errors and warnings when compiling with recent versions of `g++`, though you may need to add `EXTRA_FLAGS=-Wno-enum-compare` to avoid all warnings.

0.12.8 release - 5/6/12

- Fixed a bug that would sometimes cause an immediate segmentation fault in `--sam` mode.
- Fixed `make_galGal3.sh` script to not omit chromosome 25.
- Removed `-B` option from usage message for `bowtie-build`; that option is not implemented.
- Fixed issue that could cause Bowtie not to compile when `BOWTIE_PTHREADS` is left undefined and `pthreads.h` is not present.
- Elaborated documentation for `-B/--offbase` option to indicate that it only affects offsets in the default output mode, not in SAM mode.

Bowtie 2 beta released - 10/16/2011

- Bowtie 2 2.0.0-beta2 is available now.
- Differences between Bowtie 2 and Bowtie 1 include:
  - For reads longer than about 50 bp Bowtie 2 is generally faster, more sensitive, and uses less memory than Bowtie 1. For relatively short reads (e.g. less than 50 bp) Bowtie 1 is sometimes faster and/or more sensitive.

Site Map

Home
News archive
Getting started
Manual
Tools that use Bowtie

Latest Release

Bowtie 0.12.9 12/16/12

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

Bowtie 2: Fast, accurate read alignment
Crossbow: Genotyping, cloud computing
Tophat: RNA-Seq splice junction mapper
Cufflinks: Isoform assembly, quantitation
Myrna: Cloud, differential gene expression
Other tools using Bowtie

Pre-built indexes

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

bowtie

Open Access

Software

Ultrafast and memory-efficient alignment of short DNA sequences to the human genome

Ben Langmead, Cole Trapnell, Mihai Pop and Steven L Salzberg

Address: Center for Bioinformatics and Computational Biology, Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA.

Correspondence: Ben Langmead. Email: langmead@cs.umd.edu

Published: 4 March 2009

Genome **Biology** 2009, **10**:R25 (doi:10.1186/gb-2009-10-3-r25)

The electronic version of this article is the complete one and can be found online at http://genomebiology.com/2009/10/3/R25

Received: 21 October 2008
Revised: 19 December 2008
Accepted: 4 March 2009

Abstract

Bowtie is an ultrafast, memory-efficient alignment program for aligning short DNA sequence reads to large genomes. For the human genome, Burrows-Wheeler indexing allows Bowtie to align more than 25 million reads per CPU hour with a memory footprint of approximately 1.3 gigabytes. Bowtie extends previous Burrows-Wheeler techniques with a novel quality-aware backtracking algorithm that permits mismatches. Multiple processor cores can be used simultaneously to achieve even greater alignment speeds. Bowtie is open source http://bowtie.cbcb.umd.edu.

bowtie

The `bowtie-build` indexer

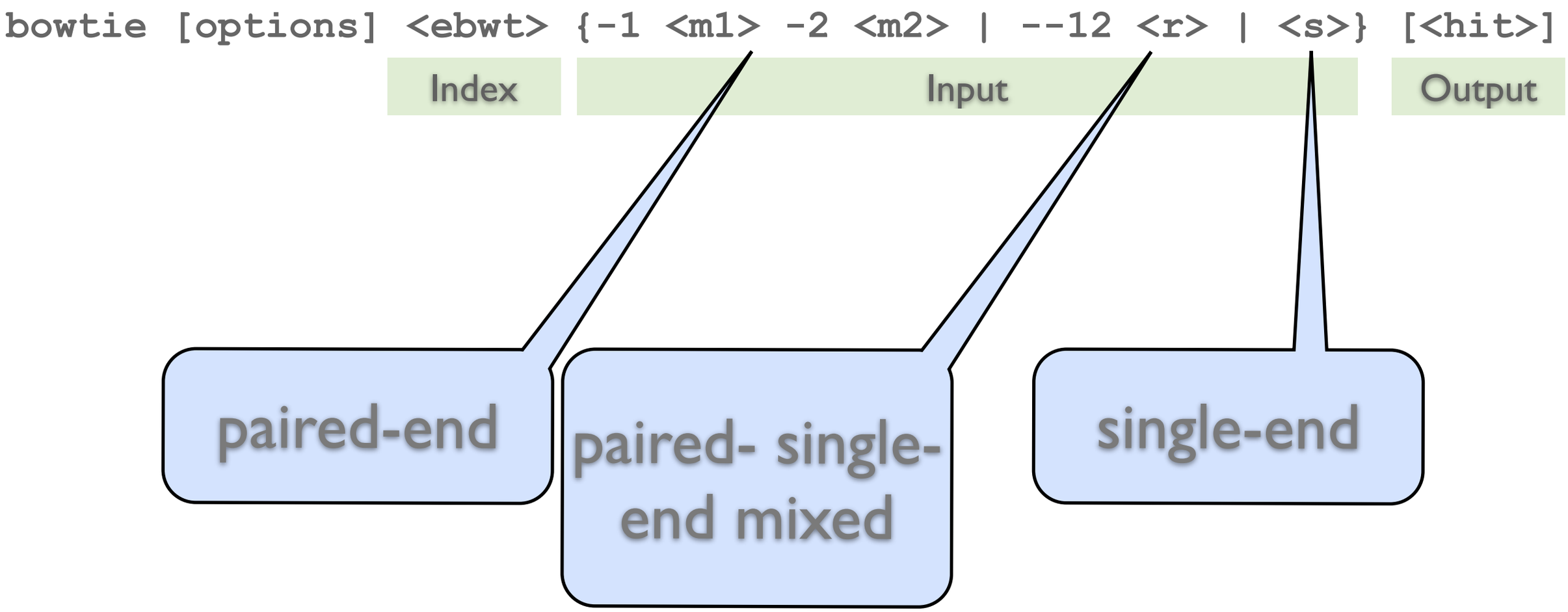
- `bowtie-build` builds a Bowtie index from a set of DNA sequences.
- `bowtie-build` outputs a set of 6 files with suffixes
  - `.1.ebwt`,
  - `.2.ebwt`,
  - `.3.ebwt`,
  - `.4.ebwt`,
  - `.rev.1.ebwt`,
  - `.rev.2.ebwt`.

- 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.

- **`bowtie-build -f human.fasta human`**

bowtie

The `bowtie` aligner



bowtie

The `bowtie` aligner

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

Index | Input | Output

`options`

- Input
- Alignment
- Reporting
- Output
- SAM
- Performance

bowtie

options

- Input
  - -q, reads in fastq format
  - -f, fasta input in fasta format
  - -5 <int>, trims <int> bases from the 5’ end
  - -3 <int>, trims <int> bases from the 3’ end

- Alignment
- Reporting
- Output
- SAM
- Performance

bowtie

`options`

- Input
- Alignment
  - **-v** <int>, Report alignments with at most <int> mismatches; **mutually exclusive with -n**
  - **-n** <int>, Maximum number of mismatches permitted in the "seed"; 0, 1, **2**, 3
  - **-e** <int>, Maximum permitted total of quality values at *all* mismatched positions; **70**
  - **-l** <int>, The "seed length"; **28**
  - -I <int>, The minimum insert size for valid paired-end alignments; **0**
  - -X <int>, The maximum insert size for valid paired-end alignments; **250**
- Reporting
- Output
- SAM
- Performance

bowtie

options

- Input
- Alignment
- Reporting
  - -k <int>, Report up to <int> valid alignments per read or pair; I
  - -a, Report all valid alignments per read or pair
  - -m <int>, Suppress all alignments for a particular read or pair if more than <int> reportable alignments exist for it
  - --best, Make Bowtie guarantee that reported singleton alignments are "best"
- Output
- SAM
- Performance

bowtie

`options`

- Input
- Alignment
- Reporting
- Output
  - --all <fileame>, Write all reads for which at least one alignment was reported to a file with name <filename>
  - --un <filename>, Write all reads that could not be aligned to a file with name <filename>.
- SAM
- Performance

bowtie

options

- Input
- Alignment
- Reporting
- Output
- SAM
  - --S, Print alignments in SAM format.
- Performance

bowtie

`options`

- Input
- Alignment
- Reporting
- Output
- SAM
- Performance
  - -p, Launch <int> parallel search threads; I

Burrows-Wheeler-Aligner

Burrows-Wheeler Aligner

Home

Introduction

Burrows-Wheeler Aligner (BWA) is an efficient program that aligns relatively short nucleotide sequences against a long reference sequence such as the human genome. It implements two algorithms, bwa-short and BWA-SW. The former works for query sequences shorter than 200bp and the latter for longer sequences up to around 100kbp. Both algorithms do gapped alignment. They are usually more accurate and faster on queries with low error rates. Please see the BWA manual page for more information.

FAQ

How can I cite BWA?

The short read alignment component (bwa-short) has been published:

Li H. and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics, 25:1754-60. [PMID: 19451168]

If you use BWA-SW, please cite:

Li H. and Durbin R. (2010) Fast and accurate long-read alignment with Burrows-Wheeler Transform. Bioinformatics, Epub. [PMID: 20080505]

(See also Errata below for a minor correction to the formulae in these papers.)

Does BWA align 454 reads?

Yes and no. The BWA-SW component of BWA works well on 454 reads about 200bp or longer. It achieves similar alignment accuracy to SSAHA2 while much faster. BWA-SW also works for shorter reads, but the sensitivity is lower. In addition, BWA-SW does not support paired-end alignment.

BWA:

- SF project page
- SF download page
- Mailing list
- BWA maual page
- Repository

Links:

- SAMtools
- MAQ

Burrows-Wheeler-Aligner

- ☑ Fast and moderate memory footprint (<4GB)
- ☑ SAM output by default
- ☑ Gapped alignment for both SE and PE reads
- ☑ Effective pairing to achieve high alignment accuracy; suboptimal hits considered in pairing.
- ☑ Limited number of errors (2 for 32bp, 4 for 100 bp, ...)
- ☑ The default configuration works for most typical input.
  - Automatically adjust parameters based on read lengths and error rates.
  - Estimate the insert size distribution on the fly

Burrows-Wheeler-Aligner

BIOINFORMATICS ORIGINAL PAPER Vol. 25 no. 14 2009, pages 1754–1760
doi:10.1093/bioinformatics/btp324

Sequence analysis

Fast and accurate short read alignment with Burrows–Wheeler transform

Heng Li and Richard Durbin*

Wellcome Trust Sanger Institute, Wellcome Trust Genome Campus, Cambridge, CB10 1SA, UK

Received on February 20, 2009; revised on May 6, 2009; accepted on May 12, 2009

Advance Access publication May 18, 2009

Associate Editor: John Quackenbush

Burrows-Wheeler-Aligner

bwa

- ☑ index (indexing of the reference)
  - bwa index -a bwtsw db.fasta
- ☑ aln (actual alignment of the short reads)
  - bwa aln db.fasta short_read.fastq > aln_sa.sai
- ☑ samse (creating of SAM output of the single reads)
  - bwa samse db.fasta aln_sa.sai short_read.fastq > aln.sam
- ☑ sampe (creating of SAM output of pair-end reads)
  - bwa sampe database.fasta aln_sa1.sai aln_sa2.sai read1.fq read2.fq > aln.sam
- ☑ bwasw (actual alignment of the long reads)
  - bwa bwasw database.fasta long_read.fastq > aln.sam
  - bwa bwasw database.fasta long_read1.fastq long_read2.fastq > aln.sam

Burrows-Wheeler-Aligner

bwa index

```
index    bwa index [-p prefix] [-a algoType] [-c] <in.db.fasta>

         Index database sequences in the FASTA format.

         OPTIONS:

-c       Build color-space index. The input fast should be in nucleotide space.

-p STR   Prefix of the output database [same as db filename]

-a STR   Algorithm for constructing BWT index. Available options are:

         is     IS linear-time algorithm for constructing suffix array. It
                requires 5.37N memory where N is the size of the database. IS is
                moderately fast, but does not work with database larger than 2GB.
                IS is the default algorithm due to its simplicity. The current
                codes for IS algorithm are reimplemented by Yuta Mori.

         bwtsw  Algorithm implemented in BWT-SW. This method works with the whole
                human genome, but it does not work with database smaller than
                10MB and it is usually slower than IS.
```

Burrows-Wheeler-Aligner

bwa aln

aln bwa aln [-n maxDiff] [-o maxGapO] [-e maxGapE] [-d nDelTail] [-i nIndelEnd] [-k maxSeedDiff] [-l seedLen] [-t nThrds] [-cRN] [-M misMsc] [-O gapOsc] [-E gapEsc] [-q trimQual] <in.db.fasta> <in.query.fq> > <out.sai>

Find the SA coordinates of the input reads. Maximum *maxSeedDiff* differences are allowed in the first *seedLen* subsequence and maximum *maxDiff* differences are allowed in the whole sequence.

OPTIONS:

-n NUM Maximum edit distance if the value is INT, or the fraction of missing alignments given 2% uniform base error rate if FLOAT. In the latter case, the maximum edit distance is automatically chosen for different read lengths. [0.04]

-o INT Maximum number of gap opens [1]

-e INT Maximum number of gap extensions, -1 for k-difference mode (disallowing long gaps) [-1]

-d INT Disallow a long deletion within INT bp towards the 3'-end [16]

-i INT Disallow an indel within INT bp towards the ends [5]

-l INT Take the first INT subsequence as seed. If INT is larger than the query sequence, seeding will be disabled. For long reads, this option is typically ranged from 25 to 35 for '-k 2'. [inf]

-k INT Maximum edit distance in the seed [2]

-t INT Number of threads (multi-threading mode) [1]

-M INT Mismatch penalty. BWA will not search for suboptimal hits with a score lower than (bestScore-misMsc). [3]

-O INT Gap open penalty [11]

-E INT Gap extension penalty [4]

-R INT Proceed with suboptimal alignments if there are no more than INT equally best hits. This option only affects paired-end mapping. Increasing this threshold helps to improve the pairing accuracy at the cost of speed, especially for short reads (~32bp).

-c Reverse query but not complement it, which is required for alignment in the color space.

-N Disable iterative search. All hits with no more than *maxDiff* differences will be found. This mode is much slower than the default.

-q INT Parameter for read trimming. BWA trims a read down to argmax_x{\sum_{i=x+1}^l(INT-q_i)} if q_l<INT where l is the original read length. [0]

Burrows-Wheeler-Aligner

bwa samse

bwa sampe

samse bwa samse [-n maxOcc] <in.db.fasta> <in.sai> <in.fq> > <out.sam>

Generate alignments in the SAM format given single-end reads. Repetitive hits will be randomly chosen.

OPTIONS:

-n INT Maximum number of alignments to output in the XA tag for reads paired properly. If a read has more than INT hits, the XA tag will not be written. [3]

sampe bwa sampe [-a maxInsSize] [-o maxOcc] [-n maxHitPaired] [-N maxHitDis] [-P] <in.db.fasta> <in1.sai> <in2.sai> <in1.fq> <in2.fq> > <out.sam>

Generate alignments in the SAM format given paired-end reads. Repetitive read pairs will be placed randomly.

OPTIONS:

-a INT Maximum insert size for a read pair to be considered being mapped properly. Since 0.4.5, this option is only used when there are not enough good alignment to infer the distribution of insert sizes. [500]

-o INT Maximum occurrences of a read for pairing. A read with more occurrneces will be treated as a single-end read. Reducing this parameter helps faster pairing. [100000]

-P Load the entire FM-index into memory to reduce disk operations (base-space reads only). With this option, at least 1.25N bytes of memory are required, where N is the length of the genome.

-n INT Maximum number of alignments to output in the XA tag for reads paired properly. If a read has more than INT hits, the XA tag will not be written. [3]

-N INT Maximum number of alignments to output in the XA tag for disconcordant read pairs (excluding singletons). If a read has more than INT hits, the XA tag will not be written. [10]

Burrows-Wheeler-Aligner

bwa bwasw

bwasw bwa bwasw [-a matchScore] [-b mmPen] [-q gapOpenPen] [-r gapExtPen] [-t nThreads] [-w bandWidth] [-T thres] [-s hspIntv] [-z zBest] [-N nHspRev] [-c thresCoef] <in.db.fasta> <in.fq>

Align query sequences in the <in.fq> file.

OPTIONS:

-a INT Score of a match [1]

-b INT Mismatch penalty [3]

-q INT Gap open penalty [5]

-r INT Gap extension penalty. The penalty for a contiguous gap of size k is q+k*r. [2]

-t INT Number of threads in the multi-threading mode [1]

-w INT Band width in the banded alignment [33]

-T INT Minimum score threshold divided by a [37]

-c FLOAT Coefficient for threshold adjustment according to query length. Given an l-long query, the threshold for a hit to be retained is a*max{T,c*log(l)}. [5.5]

-z INT Z-best heuristics. Higher -z increases accuracy at the cost of speed. [1]

-s INT Maximum SA interval size for initiating a seed. Higher -s increases accuracy at the cost of speed. [3]

-N INT Minimum number of seeds supporting the resultant alignment to skip reverse alignment. [5]

SOAPaligner (SOAP2)

SOAP

Short Oligonucleotide Analysis Package

SOAP-popIndel | SOAPdenovo-Trans | SOAPdenovo | About

Home | SOAPindel | SOAPsnv | SOAPsv | SOAPfusion | SOAP3 | SOAPfuse | SOAPaligner | SOAPsplice | SOAPsnp

SOAPaligner

- Introduction
- System Requirements
- Download
- Installation
- Command Option
- Evaluation
- Future Development
- Acknowledgements
- SOAP 1

Feedback:
soap@genomics.org.cn

Introduction

SOAPaligner/soap2 is a member of the **SOAP (Short Oligonucleotide Analysis Package)**. It is an updated version of SOAP software for short oligonucleotide alignment. The new program features in super fast and accurate alignment for huge amounts of short reads generated by Illumina/Solexa Genome Analyzer. Compared to soap v1, it is one order of magnitude faster. It require only 2 minutes aligning one million single-end reads onto the human reference genome. Another remarkable improvement of SOAPaligner is that it now supports a wide range of the read length.

SOAPaligner benefitted in time and space efficiency by a revolution in the basic data structures and algorithms used.The core algorithms and the indexing data structures (2way-BWT) are developed by the algorithms research group of the Department of Computer Science, the University of Hong Kong (T.W. Lam, Alan Tam, Simon Wong, Edward Wu and S.M. Yiu).

System Requirements

1. Hardware:
a) 64-bit x86-64 CPUs with SSE instructions.
b) 8 GB main memory (for a genome as large as human's).
c) 8 GB hard disk (for a genome as large as human's).
2. Software:
a) 64-bit Linux system (kernel >=2.6).

Download

NOTE: Due to the copyright about some parts of source code, in current version, we can not open the SOAPaligner/soap2's source code. If you want to use SOAPaligner/soap2 in other platforms, please feel free to contact us and you need to show your CPU architecture and OS kernerl version. And because the data structure is incompatible with 32bit systems, we will NOT provide relevant version for you.

SOAPaligner (SOAP2)

- [x] 2bwt-builder (creates index of the reference sequence)
  - 2bwt-builder peach_chloro.fasta

- [x] soap (aligner mapping the read)
  - single reads: soap –a <reads_a> -D <index.files> -o <output></output>
  - paired-end reads: soap –a <reads_a> -b <reads_b> -D <index.files> -o <PE_output> -2 <SE_output> -m <min_insert_size> -x <max_insert_size>

SOAPaligner (SOAP2)

```
-D   STR   Prefix name for reference index [*.index].
-a   STR   Query file, for SE reads alignment or one end of PE reads
-b   STR   Query b file, one end of PE reads
-o   STR   Output file for alignment results
-2   STR   Output file contains mapped but unpaired reads when do PE alignment
-u   STR   Output file for unmapped reads, [none]
-m   INT   Minimal insert size INT allowed for PE, [400]
-x   INT   Maximal insert size INT allowed for PE, [600]
-n   INT   Filter low quality reads contain more INT bp Ns, [5]
-t         Output reads id instead reads name, [none]
-r   INT   How to report repeat hits, 0=none; 1=random one; 2=all, [1]
-R         RF alignment for long insert size(>= 2k bps) PE data, [none] FR alignment
-l   INT   For long reads with high error rate at 3'-end, those
           can't align whole length, then first align 5' INT bp
           subsequence as a seed, [256] use whole length of the read
-v   INT   Totally allowed mismatches in one read, [2]
-M   INT   Match mode for each read or the seed part of read, which
           shouldn't contain more than 2 mismaches, [4]
           0: exact match only
           1: 1 mismatch match only
           2: 2 mismatch match only
           3: [gap] (coming soon)
           4: find the best hits
-p   INT   Multithreads, n threads, [1]
```

SOAPaligner (SOAP2)

```
-D   STR   Prefix name for reference index [*.index].
-a   STR   Query file, for SE reads alignment or one end of PE reads
-b   STR   Query b file, one end of PE reads
-o   STR   Output file for alignment results
-2   STR   Output file contains mapped but unpaired reads when do PE alignment
-u   STR   Output file for unmapped reads, [none]
-m   INT   Minimal insert size INT allowed for PE, [400]
-x   INT   Maximal insert size INT allowed for PE, [600]
-n   INT   Filter low quality reads contain more INT bp Ns, [5]
-t         Output reads id instead reads name, [none]
-r   INT   How to report repeat hits, 0=none; 1=random one; 2=all, [1]
-R         RF alignment for long insert size(>= 2k bps) PE data, [none] FR alignment
-l   INT   For long reads with high error rate at 3'-end, those
           can't align whole length, then first align 5' INT bp
           subsequence as a seed, [256] use whole length of the read
-v   INT   Totally allowed mismatches in one read, [2]
-M   INT   Match mode for each read or the seed part of read, which
           shouldn't contain more than 2 mismaches, [4]
           0: exact match only
           1: 1 mismatch match only
           2: 2 mismatch match only
           3: [gap] (coming soon)
           4: find the best hits
-p   INT   Multithreads, n threads, [1]
```

SOAPaligner (SOAP2)

```
-D   STR   Prefix name for reference index [*.index].
-a   STR   Query file, for SE reads alignment or one end of PE reads
-b   STR   Query b file, one end of PE reads
-o   STR   Output file for alignment results
-2   STR   Output file contains mapped but unpaired reads when do PE alignment
-u   STR   Output file for unmapped reads, [none]
-m   INT   Minimal insert size INT allowed for PE, [400]
-x   INT   Maximal insert size INT allowed for PE, [600]
-n   INT   Filter low quality reads contain more INT bp Ns, [5]
-t         Output reads id instead reads name, [none]
-r   INT   How to report repeat hits, 0=none; 1=random one; 2=all, [1]
-R         RF alignment for long insert size(>= 2k bps) PE data, [none] FR alignment
-l   INT   For long reads with high error rate at 3'-end, those
           can't align whole length, then first align 5' INT bp
           subsequence as a seed, [256] use whole length of the read
-v   INT   Totally allowed mismatches in one read, [2]
-M   INT   Match mode for each read or the seed part of read, which
           shouldn't contain more than 2 mismaches, [4]
           0: exact match only
           1: 1 mismatch match only
           2: 2 mismatch match only
           3: [gap] (coming soon)
           4: find the best hits
-p   INT   Multithreads, n threads, [1]
```

SOAPaligner (SOAP2)

```
-D   STR   Prefix name for reference index [*.index].
-a   STR   Query file, for SE reads alignment or one end of PE reads
-b   STR   Query b file, one end of PE reads
-o   STR   Output file for alignment results
-2   STR   Output file contains mapped but unpaired reads when do PE alignment
-u   STR   Output file for unmapped reads, [none]
-m   INT   Minimal insert size INT allowed for PE, [400]
-x   INT   Maximal insert size INT allowed for PE, [600]
-n   INT   Filter low quality reads contain more INT bp Ns, [5]
-t         Output reads id instead reads name, [none]
-r   INT   How to report repeat hits, 0=none; 1=random one; 2=all, [1]
-R         RF alignment for long insert size(>= 2k bps) PE data, [none] FR alignment
-l   INT   For long reads with high error rate at 3'-end, those
           can't align whole length, then first align 5' INT bp
           subsequence as a seed, [256] use whole length of the read
-v   INT   Totally allowed mismatches in one read, [2]
-M   INT   Match mode for each read or the seed part of read, which
           shouldn't contain more than 2 mismaches, [4]
           0: exact match only
           1: 1 mismatch match only
           2: 2 mismatch match only
           3: [gap] (coming soon)
           4: find the best hits
-p   INT   Multithreads, n threads, [1]
```

SOAPaligner (SOAP2)

```
-D   STR   Prefix name for reference index [*.index].
-a   STR   Query file, for SE reads alignment or one end of PE reads
-b   STR   Query b file, one end of PE reads
-o   STR   Output file for alignment results
-2   STR   Output file contains mapped but unpaired reads when do PE alignment
-u   STR   Output file for unmapped reads, [none]
-m   INT   Minimal insert size INT allowed for PE, [400]
-x   INT   Maximal insert size INT allowed for PE, [600]
-n   INT   Filter low quality reads contain more INT bp Ns, [5]
-t         Output reads id instead reads name, [none]
-r   INT   How to report repeat hits, 0=none; 1=random one; 2=all, [1]
-R         RF alignment for long insert size(>= 2k bps) PE data, [none] FR alignment
-l   INT   For long reads with high error rate at 3'-end, those
           can't align whole length, then first align 5' INT bp
           subsequence as a seed, [256] use whole length of the read
-v   INT   Totally allowed mismatches in one read, [2]
-M   INT   Match mode for each read or the seed part of read, which
           shouldn't contain more than 2 mismaches, [4]
           0: exact match only
           1: 1 mismatch match only
           2: 2 mismatch match only
           3: [gap] (coming soon)
           4: find the best hits
-p   INT   Multithreads, n threads, [1]
```