The DIAMOND sequence aligner

Introduction

DIAMOND is a sequence aligner for protein and translated DNA searches and functions as a drop-in replacement for the NCBI BLAST software tools. It is suitable for protein-protein search as well as DNA-protein search on short reads and longer sequences including contigs and assemblies, providing a speedup of BLAST ranging up to x20,000.

If you use DIAMOND in published research, please cite B. Buchfink, Xie C., D. Huson, "Fast and sensitive protein alignment using DIAMOND", Nature Methods 12, 59-60 (2015).

Support & updates are available at `http://github.com/bbuchfink/diamond/`.

1 Quick start guide

We assume to have a protein database file in FASTA format named `nr.faa` and a file of DNA reads that we want to align named `reads.fna`. In order to set up a reference database for DIAMOND, the makedb command needs to be executed with the following command line:

```
diamond makedb --in nr.faa -d nr
```

This will create a binary DIAMOND database file with the specified name (`nr.dmnd`). The alignment task may then be initiated using the `blastx` command like this:

```
diamond blastx -d nr -q reads.fna -o matches.m8
```

The output file here is specified with the `-o` option and named `matches.m8`. By default, it is generated in BLAST tabular format.

Note:

- The program may use quite a lot of memory and also temporary disk space. Should the program fail due to running out of either one, you need to set a lower value for the block size parameter `-b` (see 3.6).
- The default (fast) mode was mainly designed for short reads. For longer sequences, the sensitive modes (options `--sensitive` or `--more-sensitive`) are recommended.
- The full speed of the program is only attained on large query datasets. It is currently not efficient in mapping a smaller number of query sequences.
- The default e-value cutoff of DIAMOND is 0.001 while that of BLAST is 10, so by default the program will search a lot more stringently than BLAST and not report weak hits.

2 Installation & requirements

DIAMOND compiles as generic C++ code and has no particular requirements on the hardware architecture, but it makes use of the SSE instruction set of the Intel/AMD x86-64 platform when

available and will run considerably faster on that platform. It runs on POSIX-compatible operating systems (Linux, FreeBSD, OS X) as well as on Microsoft Windows.

A high-memory server is recommended for better performance, but the program can be run on standard desktop computers.

DIAMOND is actively developed software, so it is recommended to always use the latest version of the program.

Compiled binaries are available for download for Linux and Windows. Users of Max OS X need to compile the software from source.

2.1 Binary download

2.1.1 Linux

A precompiled binary is available for recent Linux systems and may be downloaded for immediate use:

```
wget http://github.com/bbuchfink/diamond/releases/download/v0.8.28/diamond-linux64.tar.gz
tar xzf diamond-linux64.tar.gz
```

If the binary does not work on your system, you need to compile the software from source.

2.1.2 FreeBSD

On FreeBSD, you can use `pkg install diamond` to install the software.

2.1.3 Windows

A binary executable for Windows can be downloaded at the DIAMOND github page. You also need to install the Visual C++ Redistributable.

2.2 Compiling from source

Compilation requires GCC 4.1 or later, CMake 2.6 or later as well as libpthread and zlib including development headers. To compile DIAMOND from source, invoke the following commands on the shell:

```
wget http://github.com/bbuchfink/diamond/archive/v0.8.28.tar.gz
tar xzf v0.8.28.tar.gz
cd diamond-0.8.28
mkdir bin
cd bin
cmake ..
make install
```

Note:

- Use `cmake -DCMAKE_INSTALL_PREFIX=...` to install to a different prefix.
- Use `cmake -DBUILD_STATIC=ON` to create a statically linked executable.

3 Command line options

3.1 Commands

Commands are issued as the first parameter on the command line and set the task to be run by the program.

`makedb`
: Create a DIAMOND formatted reference database from a FASTA input file.

`blastp`
: Align protein query sequences against a protein reference database.

`blastx`
: Align translated DNA query sequences against a protein reference database.

`view`
: Generate formatted output from DAA files.

`version`
: Print version information.

3.2 Makedb options

`--in <file>`
: Path to the input protein reference database file in FASTA format (may be gzip compressed).

`--db/-d <file>`
: Path to the output DIAMOND database file.

3.3 General & IO options

`--threads/-p #`
: Number of CPU threads. By default, the program will auto-detect and use all available virtual cores on the machine.

`--db/-d <file>`
: Path to the DIAMOND database file.

`--query/-q <file>`
: Path to the query input file in FASTA or FASTQ format (may be gzip compressed).

`--query-gencode #`
: Genetic code used for translation of query in BLASTX mode. A list of possible values can be found at the NCBI website. By default, the Standard Code is used.

`--out/-o <file>`
: Path to the output file. If this parameter is omitted, the results will be written to the standard output and all other program output will be suppressed.

`--outfmt/-f #`

Format of the output file. The following values are accepted:

`0` BLAST pairwise format.

`5` BLAST XML format.

`6` BLAST tabular format (default). This format can be customized, the `6` may be followed by a space-separated list of the following keywords, each specifying a field of the output.

qseqid Query Seq - id
qlen Query sequence length
sseqid Subject Seq - id
sallseqid All subject Seq - id(s), separated by a ';'
slen Subject sequence length
qstart Start of alignment in query
qend End of alignment in query
sstart Start of alignment in subject
send End of alignment in subject
qseq Aligned part of query sequence
sseq Aligned part of subject sequence
evalue Expect value
bitscore Bit score
score Raw score
length Alignment length
pident Percentage of identical matches
nident Number of identical matches
mismatch Number of mismatches
positive Number of positive - scoring matches
gapopen Number of gap openings
gaps Total number of gaps
ppos Percentage of positive - scoring matches
qframe Query frame
btop Blast traceback operations(BTOP)
stitle Subject Title
salltitles All Subject Title(s), separated by a '<>'
qcovhsp Query Coverage Per HSP
qtitle Query title

By default, there are 12 preconfigured fields: `qseqid sseqid pident length mismatch gapopen qstart qend sstart send evalue bitscore`.

`100` DIAMOND alignment archive (DAA). The DAA format is a proprietary binary format that can subsequently be used to generate other output formats using the `view` command. It is also supported by MEGAN and allows a quick import of results.

`101` SAM format.

`--compress (0,1)`

Enable compression of the output file. 0 (default) means no compression, 1 means gzip compression.

3.4 Sensitivity & speed options

`--sensitive`
: This mode is a lot more sensitive that the default and generally recommended for aligning longer sequences. The default mode is mainly designed for short read alignment, i.e. finding significant matches of >50 bits on 30-40aa fragments.

`--more-sensitive`
: This mode provides some additional sensitivity compared to the sensitive mode.

`--band #`
: Dynamic programming band for seed extension. This corresponds to the maximum length of gaps that can be found in alignments.

3.5 Scoring & reporting options

`--gapopen #`
: Gap open penalty (default=11).

`--gapextend #`
: Gap extension penalty (default=1).

`--matrix <matrix name>`
: Scoring matrix. The following matrices are supported, with the default being BLOSUM62.

Matrix	Supported values for (gap open)/(gap extend)
BLOSUM45	(10-13)/3; (12-16)/2; (16-19)/1
BLOSUM50	(9-13)/3; (12-16)/2; (15-19)/1
BLOSUM62	(6-11)/2; (9-13)/1
BLOSUM80	(6-9)/2; 13/2; 25/2; (9-11)/1
BLOSUM90	(6-9)/2; (9-11)/1
PAM250	(11-15)/3; (13-17)/2; (17-21)/1
PAM70	(6-8)/2; (9-11)/1
PAM30	(5-7)/2; (8-10)/1

`--seg (yes,no)`
: Enable SEG masking of low complexity segments in the query. The default is no for blastp and yes for blastx.

`--max-target-seqs #`
: The maximum number of target sequences per query to report alignments for (default=25).

`--top #`
: Report alignments within the given percentage range of the top alignment score for a query (overrides `--max-target-seqs` option).

`--evalue #`
: Maximum expected value to report an alignment (default=0.001).

`--min-score #`
: Minimum bit score to report an alignment. Setting this option will override the `--evalue` parameter.

`--id #`
Report only alignments above the given percentage of sequence identity.

`--query-cover #`
Report only alignments above the given percentage of query cover.

`--single-domain`
In general more than one HSP may exist for the same query/subject pair, and by default the program will report all HSPs that were found. This option will suppress the reporting of lower scoring HSPs and only report the single best HSP for each query/subject pair.

3.6 Memory & performance options

`--block-size/-b #`
Block size in billions of sequence letters to be processed at a time. This is the main parameter for controlling the program's memory usage. Bigger numbers will increase the use of memory and temporary disk space, but also improve performance. The program can be expected to roughly use six times this number of memory (in GB). So for the default value of `-b2.0`, the memory usage will be about 12 GB.

`--tmpdir/-t <directory>`
Directory to be used for temporary storage. This is set to the output directory by default. The amount of disk space that will be used depends on the program's settings and your data. As a general rule you should ensure that 100 GB of disk space are available here. If you run the program in a cluster environment, and disk space is only available over a slow network based file system, you may want to set the `--tmpdir` option to `/dev/shm`. This will keep temporary information in memory and thus increase the program's memory usage substantially.

`--index-chunks/-c #`
The number of chunks for processing the seed index (default=4). This option can be additionally used to tune the performance. It is recommended to set this to 1 on a high memory server, which will increase performance and memory usage, but not the usage of temporary disk space.

3.7 View options

`--daa/-a <file>`
Path to input file in DAA format.

`--out/-o <file>`
Path to output file. If this parameter is omitted, the results will be written to the standard output and all other program output will be suppressed.

`--outfmt/-f #`
Format of the output file, see 3.3.

`--compress (0,1)`
Enable compression of the output file, see 3.3.

4 Optimizing performance

You should take the time to configure the program to your needs instead of just running it with default settings, as this will substantially optimize the use of system resources. Some points to consider:

- Set the `-e` parameter (maximum expected value) as low as possible.
- Set the `-k` parameter (number of target sequences to report alignments for) as low as possible. This will improve performance and reduce the use of temporary disk space and the size of the output files.
- Consider using the `--top` parameter instead of `-k`. For example, setting `--top 5` will report all alignments whose score is at most 5% lower than the top alignment score for a query.
- Configure the tabular output format to only report the output fields that are actually required (see 3.3.).
- Use the BLAST tabular format for machine-based processing of the output. Use the XML and SAM format only if required by your downstream tools or if you prefer it as a human-readable format.
- Use the option `--compress 1` to automatically generate gzip-compressed output files.

About

Bug reports, support and other inquiries may be sent to Benjamin Buchfink.