CD-HIT AuxTools: Manual

Last updated: 2012-03-08
http://cd-hit.org
http://bioinformatics.org/cd-hit/
Program developed by Weizhong Li's lab at UCSD
http://weizhong-lab.ucsd.edu liwz@sdsc.edu

Contents

1 Introduction

CD-HIT AuxTools is a set of auxiliary programs that can be used to assist the analysis of the next generation sequencing data. It currently includes programs for removing read duplicates, finding pairs of overlapping reads or joining pair-end reads etc.

2 cd-hit-dup

cd-hit-dup is a simple tool for removing duplicates from sequencing reads, with optional step to detect and remove chimeric reads. A number of options are provided to tune how the duplicates are removed. Running the program without arguments should print out the list of available options, as the following:

```
Options:
    -i         Input file;
    -i2        Second input file;
    -o         Output file;
    -d         Description length (default 0, truncate at the first
       whitespace character)
    -u         Length of prefix to be used in the analysis (default 0,
       for full/maximum length);
    -m         Match length (true/false, default true);
    -e         Maximum number/percent of mismatches allowed;
    -f         Filter out chimeric clusters (true/false, default false)
       ;
    -s         Minimum length of common sequence shared between a
       chimeric read
               and each of its parents (default 30, minimum 20);
    -a         Abundance cutoff (default 1 without chimeric filtering,
       2 with chimeric filtering);
    -b         Abundance ratio between a parent read and a chimeric
       read (default 1);
    -p         Dissimilarity control for chimeric filtering (default 1)
       ;
```

2.1 Option details

2.2 Common options

Here are the more detailed description of the options.

```
    -i         Input file;
```

Input file that must be in fasta or fastq format.

```
    -i2        Second input file;
```

cd-hit-dup can take a pair of files as inputs, assuming they contain sequences of pair-end reads. "-i" can be used to specify the file for the first end; and "-i2" can be used to specify the file for the second end.

When two files of pair-end reads are used as inputs, each pair of reads will be concatenated into a single one. And the following steps of duplicate and chimeric detection and removing.

```
    -o        Output file;
```

Output file which contains a list of reads without duplicates.

```
    -d        Description length (default 0, truncate at the first
       whitespace character)
```

The length of description line that should be written to the output.

```
    -u        Length of prefix to be used in the analysis (default 0,
       for full/maximum length);
```

For pair-end inputs, the program will take part (whole or prefix) of the first end and part (whole or prefix) of the second read, and join them together to form a single read to do the analysis. A positive value of this option specifies the length of the prefix to be taken from each read. If a read is shorter than this length, letter 'N's will be appended to the read to make up for the length. When this option is not used or is used with a non-positive value, the program will use the length of the longest read as the value of this option.

For single input analysis, only a positive value of this option will be effective. It also allows the program to use only the prefix up to the specified length of each read to do the analysis. In case that a read is shorter than this length, no 'N' is appended to the read since it is not necessary.

2.3 Options for duplicate detection

```
    -m        Match length (true/false, default true);
```

"-m" specifies whether the lengths of two reads should be exactly the same to be considered as duplicates.

```
    -e        Maximum number/percent of mismatches allowed;
```

Maximum number/percent of mismatches can be specified to control the similarity between two reads for duplicate and chimeric detection. For duplicate detection, any two reads with number of mismatches no greater than the specified value are considered to be duplicates. For chimeric detection, this option control how similar a read should be to either of its parents.

2.4 Options for chimeric filtering

```
    -f         Filter out chimeric clusters (true/false, default false)
    ;
```

This option specifies whether or not to carry out an additional step to filter out chimeric clusters.

```
    -s         Minimum length of common sequence shared between a
    chimeric read
               and each of its parents (default 30, minimum 20);
```

A read or cluster representative is considered as a potential chimeric only if it shares at least the number of bases specified by this option with either of its parents. This option is effective only if the option is set to true for filtering chimeric clusters.

```
    -a         Abundance cutoff (default 1 without chimeric filtering,
    2 with chimeric filtering);
```

Each read is associated with an abundance number, which is the number of duplicates for the read. cd-hit-dup always assumes the input contains duplicates and perform the duplicate detection step. If no duplicate is found, the input is assumed to have duplicates remove in advance, and then, the program will try to obtain the abundance information from the descriptions of the reads, it interprets the number following "_abundance_" as the abundance number.

The abundance cutoff is mainly used for chimeric filtering to skip chimeric checking on reads with abundance below this cutoff.

```
    -b         Abundance ratio between a parent read and a chimeric
    read (default 1);
```

This option specifies the abundance ratio between a parent read and a chimeric read. So for a read to be chimeric, either of its parents must have abundance at least as high as the ratio times the abundance of the chimeric read.

```
    -p         Dissimilarity control for chimeric filtering (default 1)
    ;
```

Internally dissimilarity is measured by percent of mismatches with ungapped alignments. By default the percentage cutoff is set to 0.01 (one percent). This option specifies a multiplier to this percentage cutoff. A higher value will increase the dissimilarity thresholds in chimeric filtering.

2.5 Output files

cd-hit-dup will output three files. Two of them are the same as the output files of CD-HIT: one (named exactly the same as the file name specified by the "-o" option) is the cluster (or duplicate) representatives, the other is the clustering file (xxx.clstr) relating each duplicate to its representative. The third file (xxx2.clstr) contains the chimeric clusters. In this file, the description for each chimeric cluster contains cluster ids of its parent clusters from the clustering file xxx.clstr.

2.6 Examples

2.7 Duplicate Detection

Remove duplicates using default parameters:

```
cd-hit-dup -i input.fa -o output
```

By default, only reads that are identical are considered as duplicates. If "-m" is set to false, duplicates will be allowed to have different length, but the longer ones must have a prefix that is identical to the shorter ones.

Remove duplicates with a few mismatches:

```
cd-hit-dup -i input.fa -o output -e 2
cd-hit-dup -i input.fa -o output -e 0.01
```

The former will allow each duplicate read to have up to 2 mismatches when aligned to its representative; and the later will allow up to one percent mismatches.

Remove duplicates from pair-end reads:

```
cd-hit-dup -i pair-end1.fa -i2 pair-end2.fa -o output
```

Each read from "pair-end1.fa" and "pair-end2.fa" will be joint to form a single read to detect duplicates. If they all are of the same length, the full length of each ends will be used in forming the single read; otherwise, the default value of option "-u" will be used to determine how the single read is created.

Remove duplicates from pair-end reads with control on how the pair-ends are jointed:

```
cd-hit-dup -i pair-end1.fa -i2 pair-end2.fa -o output -u 100
```

With explicit "-u" options, any reads shorter than 100 will be padded with 'N's, and the longer ones will be cut down to 100 base long. Then each pair of the 100 base long reads will be jointed to form a single 200 base long read.

2.8 Chimeric Filtering

cd-hit-dup offers a very efficient way to detect chimeric reads. The basic idea is to find two parent reads whose cross-over is sufficient similar to the chimeric read, while each single parent is sufficiently dissimilar to it.

Such dissimilarity is measured by the percent of mismatches for no-gapped alignments. For a given percentage "p" (from option "-p"), a chimeric read must share at least "p" percent mismatches with any other single read, namely, it much be sufficiently dissimilar to any single read.

For more robust detection of chimeric reads, a background percentage "p_bg" is calculated as the mismatch percentage shared between the candidate chimeric read and the single read that is most similar to the candidate. If "p_bg" is greater than "1.5*p", "1.5*p" will be used as "p_bg" instead.

For a read to be classified as chimeric read, there must exist two reads/parents such that, the leading part of the read is sufficiently similar to one parent, and the rest is sufficiently similar to the other parent, with at most "p+p_bg" percent of mismatches in each part. And the crossover between the two parents must share at most "p_bg" mismatches with the chimeric read.

Chimeric filtering with default parameters:

```
cd-hit-dup -i input.fa -o output -f true
```

Chimeric filtering with specified similarity level:

```
cd-hit-dup -i input.fa -o output -f true -p 1.5
```

Chimeric filtering with specified abundance difference:

```
cd-hit-dup -i input.fa -o output -f true -a 2
```

which means each parent of a chimeric read must be a least as twice abundant as the chimeric read.

Chimeric filtering will produce a cluster file named like "xxx2.clstr", in which each cluster entry is a chimeric read/cluster. For example,

```
......
>Cluster 4 chimeric_parent1=2,chimeric_parent2=8
0   256nt, >FV9NWLF01CRIR3_abundance_23... *
```

```
>Cluster 5 chimeric_parent1=2,chimeric_parent2=0
0   250nt, >FV9NWLF01B4TBX_abundance_21... *
......
```

here "Cluster 5" contains a chimeric read "FV9NWLF01B4TBX", whose parents are identified by cluster numbers "2" and "0" from the associated "xxx.clstr" file,

```
>Cluster 0
0   252nt, >FV9NWLF01ANLX2_abundance_2239... *
>Cluster 1
0   246nt, >FV9NWLF01C3KOB_abundance_1465... *
>Cluster 2
0   260nt, >FV9NWLF01AQOWA_abundance_1284... *
......
```

So the parent reads of the chimeric read "FV9NWLF01B4TBX" are "FV9NWLF01AQ and "FV9NWLF01ANLX2".

3 cd-hit-lap

cd-hit-lap is program for extracting pairs of overlapping reads by clustering based on tail-head overlaps (with perfect matching). The basic clustering strategy is the same as that in standard CD-HIT programs. In this program, each read is clustered as either a ”representative” or a ”redundant” read. For each ”redundant” read, it must have a prefix that is identical a suffix of its representative read.

The options of this program can be obtained by running it it without any arguments:

```
[compute-0-0 cdhit-dup]$ ./cd-hit-lap
Options:
    -i        Input file;
    -o        Output file;
    -m        Minimum length of overlapping part (default 20);
    -p        Minimum percentage of overlapping part (default 0, any
       percentage);
    -d        Description length (default 0, truncate at the first
       whitespace character)
    -s        Random number seed for shuffling (default 0, no
       shuffling; shuffled before sorting by length);
    -stdout   Standard output type (default "log", other options "rep"
       , "clstr");
```

The two options ”-m” and ”-p” can be used to control the minimum overlap that is required to classify them as overlapping reads. Each pair of overlapping reads must have overlap length no less than the threshold specified by ”-m”, and must also not be less than the length threshold computed from the ”-p” option.

Since the overlapping reads are searched using a greedy strategy, so different sortings of reads may lead to different result. So it is advisable to run the program multiple times with read shuffling by different random number seeds, and then collect and merge the results.

Sometimes it may be more convenient to pipe the results of this program as stdout directly to the stdin of other programs, to do this, the option ”-stdout” can be used to choose which type (”log” for program console information, ”rep” for representative reads in FASTA or FASTQ format, ”clstr” for the clustering output in CD-HIT format) of results to be writen to the stdout.

The output format of this program is the same as the standard CD-HIT. In the .clstr file, the alignment positions indicate how the reads are overlapped. For example,

```
>Cluster 0
0   75nt, >1_lane2_624... *
1   75nt, >1_lane2_7169... at 1:65:11:75/+/100.00%
2   75nt, >1_lane2_36713... at 69:1:1:69/-/100.00%
3   75nt, >1_lane2_141482... at 1:56:20:75/+/100.00%
```

The cluster member #0 in cluster #0 is the representative of the cluster, and it overlaps with each of the other members in the cluster. For cluster member #1, "1:65:11:75/+" tells that the first 65 bases of member #1 overlaps with the last 65 bases of member #0; "69:1:1:69/-" indicates that the last 69 bases of member #2 overlaps with the first 69 bases of member #0.

4 read-linker

read-linker is a very simple program to concatenate pair-end reads into single ones. It support the following options:

```
[compute-0-0 cdhit-dup]$ ./read-linker
Options:
    -1 file        Input file, first end;
    -2 file        Input file, second end;
    -o file        Output file;
    -l number      Minimum overlapping length (default 10);
    -e number      Maximum number of errors (mismatches, default 1);
```

Only the pairs of reads that share at least a minimum overlapping length with mismatched no more than the maximum number of errors, are jointed to form a single read.