CrunchClust

An efficient algorithm for clustering 454 rRNA sequences.

Version 43 (Release)
November 24, 2011

Contact information:

Richard.Christen at unice.fr

dumaatravaie at gmail.com

Author (CrunchClust Release Version c++) : dipankar
email : dumaatravaie at gmail.com

Please do not forget to cite CrunchClust if you have used it for your work.

Introduction

Crunchclust is an efficient clustering algorithm which is capable of handling the most common Roche's **454 sequencing** error (Errors due to the presence of **Homopolymers** in the sequence). It uses **Levenshtein distance** for sequence comparison during clustering.
It also has the capability of performing strict dereplication on the raw sequence dataset and sort the raw sequences according to their abundance before doing actual clustering. In this way it reduces the dimension of the datasets, increases the clustering accuracy and reduces the computational complexity. It also provides the user the option of running it for different distance thresholds between two intervals (kmin and kmax) with a single command. Parallel version and search version of CrunchClust are under construction to meet the special needs of the scientific community.

Overview of the Algorithm

Crunchclust is a greedy incremental clustering algorithm. The first sequence from the dataset becomes the seed of the first cluster. Then, the distance between the seed and each remaining sequence is compared. If the distance of the query to the seed sequence is equal or below a given threshold then it is assigned to that cluster. Otherwise, a new cluster is defined with that sequence as the seed.

It gives user the freedom of choosing the order of their sequence, user may either arrange the sequences in their own way to cluster it with crunchclust or they can ask crunchclust to arrange them in the unique way of crunchclust , that is duplicate sequences are removed and the representatives of duplicate sequences are sorted according to their frequency before clustering. It also prints the dereplication information to a separate file that is fully compatible with the "name file" used by mothur (http://www.mothur.org/wiki/Name_file).

Levenshtein distance is a metric for measuring the amount of differences between two sequences (ie an **edit distance**). As the 454 sequences tend to start at the same place, Levenstein distance is the most appropriate measure for calculating distances between them.

Crunchclust has given the user the freedom of choice whether or not to count the **end gaps** between the sequences as distance. This is useful as the 454 sequences may or may not end at the same place.

The main benefits of **CrunchClust** are:

* Produces better quality clusters by optimal alignments using Levinstine Distance

* Portable (Works in Linux, Windows and MacOs)

* It does not require special computing power or Software installation as it runs efficiently even on ordinary laptop by using a single CPU core.

There is also no restriction of 32 bit or 64 bit version. Memory requirements is also negligible

Installation

Check first, whether your system has both **gcc** and **g++** compiler installed**.** Download the source zip file **CrunchClust.zip** and save it in a suitable folder. Extract it using unzip. Then go to that folder from command line for example :

cd /path_for_the_CrunchClust_Folder_unzipped

You can make executable file either by using the following command

make

or by simply typing the command

g++ -O3 ./CrunchClust_Version43.cpp -o ./crunchclust

Windows users can use **Cygwin** for **g++** compiler.

Alignment for distance calculation

CrunchClust computes the Levinstine distance in the following way :

Example 1 (By correcting homopolymer error) :

ATGTGGGGTAT
ATGTGGG - TAT

The difference between the above two sequences will be calculated as 0 , Because the differences in the homopolymer region (4 consecutive G in Reference sequence and three consecutive G in Comparing sequence) is not counted.

Example 2 (By not correcting homopolymer error) :

ATGTGGGGTAT
ATGTGGG - TAC

For this example, the difference between the above two sequences will be calculated as 2 , because the differences in the homopolymer region is counted and the single difference (mutation) outside the homopolymer region is also counted.

Different options for distance calculations

Crunchclust also provides two possibilities for calculating the Levinstine distance :

1) Distance calculation without considering the end gaps as distance

For example the distance between the following two sequences

ATGTGGGGTATA
ATGTGGG–TAC -

will be calculated as 2 if the user chooses the option for not counting the end gaps and also wants to count all the differences as distance. The example command for this types of distance computation during the clustering is given below :

Example Command :

./crunchclust --diff 10 --in Input_Sequence_file.fasta --out Output_file_Name.clstr --d_all --noendgaps

2) Distance calculation considering the end gaps as distance

For example the distance between the following two sequences

ATGTGGGGTATA
ATGTGGG–TAC -

will be calculated as 2 if the user chooses the option for counting the end gaps as difference along with homopolymer option for distance calculation.
The example command for this types of distance computation during the clustering is given below :

Example Command :

./crunchclust --diff 10 --in Input_Sequence_file.fasta --out Output_file_Name.clstr --d_hl –endgaps

Other useful command example

./crunchclust --in Input_Sequence_file.fasta --out output_Sequence_file.clstr --d_hl --endgaps --kmin 3 --kmax 9 --ksteps 3

The above command will cluster the input sequences **3** times with 3 different distance threashold 3, 6 and 9 and will produce 3 output cluster files named **output_Sequence_file.clstr_3**, **output_Sequence_file.clstr_6**, **output_Sequence_file.clstr_9**

Dereplication and sorting of sequences before clustering:

Crunchclust also provides the options of dereplication. With this option duplicate sequences in the raw datasets are removed and then the sequences are sorted according to their abundance in the datasets before doing the clustering. Dereplication produces better clustering accuracy and also reduces the computation complexity. The example command for doing dereplication along with the clustering is given below :

Example Command :

./crunchclust --in Input_Sequence_file.fasta --out Output_file_Name.clstr --d_all --endgaps –strict –min 10 –max 400 --kmin 0 --kmax 10 --ksteps 1

This above command will do the following things:

- It will dereplicate the Input fasta file and print the dereplication information into a file named **Input_Sequence_file_dereplication.names** which has a format like mothur **name** files http://www.mothur.org/wiki/Name_file . The first column is the tag name of representative sequence along with its frequency information at the end for example if the sequence name is **>representive_seq** and has frequency 10 then its name as a representive sequence will be **representive_seq_10** in the name file. As the **Name** file format does not allow to include **“>”** with the tag name.

- Then it will custer the dereplicated sequences **3** times with 3 different distance threashold 3, 6 and 9 and will produce 3 output cluster files named **output_Sequence_file.clstr_3**, **output_Sequence_file.clstr_6**, **output_Sequence_file.clstr_9**

Command line Options

--h : For command line help

options for dereplications :

--strict : Dereplication before clustering. Absense of this option means, crunchclust will not do any dereplication. After dereplication the frequency of the sequence in the raw data will be added at the end of the tag name of each retained sequence which

is the single representative of the duplicate sequences.

--min : Sequences shorter then this size will not be considered.
Default is 5, which means any sequence shorter then this length will be deleted

--max : Sequences bigger then this length will be cropped at this length.
Default is 400, which means any sequence bigger then this length will be cropped at 400

--keep_n : Whether to discard the sequences containing N,
If you mention this option then the sequences containing N will be kept during dereplication otherwise they will be discarded.
This option works along with the option –strict

Options for clustering :

--diff : Levenstine distance threshold in integer number like 0, 1, 2, 3, 4 . . .

--in : Input Sequence File

--out : Name of the outputfile

Options for clustering in loops with different distance threshold :

--kmin : Start clustering at this difference

--kmax : Keep clustering till this difference threshold

--ksteps : Increase distance thresehold by this value at each loop

Two options for calculating the distance : --d_hl, --d_all

--d_hl : Does not count differences in the homopolymer region,

--d_all : Counts all the differences

Options for counting the end gaps

--endgaps : For counting the end gaps
--noendgaps : For not counting the end gaps

Please do not forget to cite CrunchClust if you have used it for your work. ***For any question and huge project (very large datasets) collaboration please feel free to send us an email at the email address mentioned in the first page of this documentation.***