Repeatmodeler

Contents

Introduction

This is a sister program to RepeatMasker. It also has no publication behind it.

Usage

- NB always use **-engine ncbi** when running RepeatModeler.
- NB2 There is a warning like so in the output:

```
Use of uninitialized value $1 in string ne at /shelf/modulefiles/tools/RepeatModeler/1.0.10/SequenceSimilarityMatrix.pm line 273, <MATRIX> li
```

Because it is only a warning, it should not affect the proper running of the program. In any case, it will be repaired shortly.

Here are the contents of the help manual:

```
$ RepeatModeler --help
No database indicated

NAME
    RepeatModeler - Model repetitive DNA

SYNOPSIS
      RepeatModeler [-options] -database <XDF Database>

DESCRIPTION
    The options are:

    -h(elp)
        Detailed help

    -database
        The prefix name of a XDF formatted sequence database containing the
        genomic sequence to use when building repeat models. The database
        may be created with the WUBlast "xdformat" utility or with the
        RepeatModeler wrapper script "BuildXDFDatabase".

    -engine <abblast|wublast|ncbi>
        The name of the search engine we are using. I.e abblast/wublast or
        ncbi (rmblast version).

    -pa #
        Specify the number of shared-memory processors available to this
        program. RepeatModeler will use the processors to run BLAST searches
        in parallel. i.e on a machine with 10 cores one might use 1 core for
        the script and 9 cores for the BLAST searches by running with "-pa
        9".

    -recoverDir <Previous Output Directory>
        If a run fails in the middle of processing, it may be possible
        recover some results and continue where the previous run left off.
        Simply supply the output directory where the results of the failed
        run were saved and the program will attempt to recover and continue
        the run.

    -srand #
        Optionally set the seed of the random number generator to a known
        value before the batches are randomly selected ( using Fisher Yates
        Shuffling ). This is only useful if you need to reproduce the sample
        choice between runs. This should be an integer number.

SEE ALSO
        RepeatMasker, WUBlast

COPYRIGHT
     Copyright 2005-2017 Institute for Systems Biology

AUTHOR
     Robert Hubley <rhubley@systemsbiology.org>
     Arian Smit <asmit@systemsbiology.org>
```

Example run

Let's try Repeatmodeler on the S288C yeast reference genome. First the database must be built:

```
BuildDatabase -name yeast288c -engine ncbi S288_maniid.fsa
```

This is only a wrapper to the blast makedatabase function, and should finish quickly depending on he size of the file of course. Yeast is small (12MB), so it was over quickly.

The next operation can then be RepeatModeler itself:

```
RepeatModeler -engine ncbi -pa 8 -database yeast288c
```

Here we are running with 8 parallel processes (at a guess **-pa** is a mnemonic for parallel).

Repeatmodeler creates a sufloder int eh directory in which is is run and then goes through a series of processing rounds.

Round 1 is a repeat search using Repeatscout and its build_lmer_table. A series of subrounds follows.

Next is round 2 which uses TRFMask and when finished gives the number of HSPs and then uses RECON for family definitions and then it refines the family.

Round 3 is similar to round 2. For yeast the program finished here and produced:

- **yeast288c-families.fa** - Consensus sequences for each family identified.
- **yeast288c-families.stk** - Seed alignments for each family identified

as well as many files starting with **trf-Results-**. These are just normal output from the **trf** program, including the files ending with err**. They are not necessarily errors, but rather are output from STDERR.**

Installation notes

RepeatModeler has a number of dependencies. How these are installed are detailed below:

nseg

nseg and **nmerge** (included in **nseg**) are quite old programs coded in C by NCBI. These were installed on all machines in /**usr**/**local**/**bin**

RECON

The same is done with RECON, which consists of the following executables:

- imagespread
- eledef
- eleredef
- edgeredef
- famdef

So all these are now available in /**usr**/**local**/**bin** in the nodes as well.

RepeatScout

As this also only includes two C-coded executables with quite good names, it is easy enough to install it locally on all the nodes:

- RepeatScout
- build_lmer_table

and also the following perl scripts (using the unusual *.prl extension) must be also be installed:

- filter-stage-1.prl
- filter-stage-2.prl
- merge-lmer-tables.prl
- compare-out-to-gff.prl

RepeatModeler itself

This is quite similar to RepeatMasker, although there is an opportunity to manually edit **RepModelConfig.pm.tmp** which is more accurate and convenient than the automatic configure script. Predictably, it will only be active when the tmp extension is hived off.

Links

- the RepeatScount paper