|  |  |
| --- | --- |
| breeder 1.1 | |
|  |  |
|  | User and developer tutorial |
|  |  |
|  |  |

**Copyright and Permission Notice**

Copyright (c) 2009 Samuel Flores, Uppsala University

For full copyright and permission notice, see the *Reference guide.*

**Acknowledgments**

Ongoing development of breeder is funded by Uppsala University, Sweden.

**Table of Contents**

1 Overview 7

2 Working with subversion (svn) 13

3 Compiling 14

4 Running 15

4.1 Starting a new job 18

4.2 Useful MySQL commands: 18

# 

# Overview

breeder is a genetic algorithm based protein-protein interface optimizer.

# Installing Simbody (SimTK)

MMB requires SimTK. You can get precompiled binaries from SimTK.org. However if you’re in my lab you can just copy mine, since it doesn’t get updated with all that much frequency. First, make a directory called:

~/SimTK

Then copy my SimTK to yours:

cp –Rf ~samuelf/SimTK/\* ~/SimTK

How to update Simbody and molmodel:

cd ~/svn/simbody/trunk

svn update

cd ~/svn/molmodel/trunk

svn update

Then compile:

cd ~/svn/molmodel/trunk/build

[samuelf@tintin1 build]$ make -j8

[samuelf@tintin1 build]$ make install

# Checking out MMB with subversion (svn)

Breeder requires MMB. You can get precompiled binaries from SimTK.org. However if you’re in my lab you probably need to install from source so you can get updates easily. In that case, do the following. First, make a directory called:

~/svn/RNAToolbox/trunk

svn checkout <https://simtk.org/svn/rnatoolbox>/trunk ~/svn/rnatoolbox/trunk

If you are going to make changes to a file, it is good to make sure you have an up to date distribution:

cd ~/svn/rnatoolbox/trunk

svn update .

Let’s say you modify BiopolymerClass.cpp. The way to commit this to the repository is this:

svn commit ~/svn/ rnatoolboxtrunk/src/BiopolymerClass.cpp

If you get an error saying the editor has not been set, this means you have to issue

export SVN\_EDITOR=vim

You should add this line to your ~/.bash\_profile so you don’t do this every time.

Reference:

http://www.techchorus.net/how-set-svneditor-environment-variable-vim

# Compiling MMB

Your LD\_LIBRARY\_PATH should point to wherever you have SimTK libraries, e.g. :

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:~/SimTK/lib64

You will then need to compile:

mkdir ~/svn/RNAToolbox/trunk/build

cd ~/svn/RNAToolbox/trunk/build

ccmake ..

At this point cmake may puke and tell you a newer version of cmake is needed. I have a 2.8 version here:

/bubo/home/h17/samuelf/cmake-2.8.7-Linux-i386

Make a similar directory in your home directory and copy the contents of the above into it. Uppmax is a bit difficult with user groups, so you may find it useful to copy it first to /scratch .

You will then find it useful to create an alias which overrides the default ccmake. Put something like this in your ~/.bash\_profile :

alias ccmake='/bubo/home/h17/samuelf/cmake-2.8.7-Linux-i386/bin/ccmake'

.. except substituting your own home directory of course.

Then check your cmake variables. The big pitfalls are the ones that locate your linked libraries ( SimTK). Also, breeder should for now be compiled in Debug mode. Here are values that work for me:

BUILD\_MMB\_GUI\_TEST OFF

BUILD\_MMB\_SHARED\_EXEC ON

BUILD\_MMB\_SHARED\_LIB ON

BUILD\_MMB\_STATIC OFF

Boost\_INCLUDE\_DIR

CMAKE\_AR /usr/bin/ar

CMAKE\_BUILD\_TYPE Release

CMAKE\_COLOR\_MAKEFILE ON

CMAKE\_CXX\_COMPILER /usr/bin/c++

CMAKE\_CXX\_FLAGS

CMAKE\_CXX\_FLAGS\_DEBUG -g

CMAKE\_CXX\_FLAGS\_MINSIZEREL -Os -DNDEBUG

CMAKE\_CXX\_FLAGS\_RELEASE -O3 -DNDEBUG

CMAKE\_CXX\_FLAGS\_RELWITHDEBINFO -O2 -g

CMAKE\_C\_COMPILER /usr/bin/gcc

CMAKE\_C\_FLAGS

CMAKE\_C\_FLAGS\_DEBUG -g

CMAKE\_C\_FLAGS\_MINSIZEREL -Os -DNDEBUG

CMAKE\_C\_FLAGS\_RELEASE -O3 -DNDEBUG

CMAKE\_C\_FLAGS\_RELWITHDEBINFO -O2 -g

CMAKE\_EXE\_LINKER\_FLAGS

CMAKE\_EXE\_LINKER\_FLAGS\_DEBUG

CMAKE\_EXE\_LINKER\_FLAGS\_MINSIZE

CMAKE\_EXE\_LINKER\_FLAGS\_RELEASE

CMAKE\_EXE\_LINKER\_FLAGS\_RELWITH

CMAKE\_EXPORT\_COMPILE\_COMMANDS OFF

CMAKE\_INSTALL\_PREFIX /usr/local

CMAKE\_LINKER /usr/bin/ld

CMAKE\_MAKE\_PROGRAM /usr/bin/gmake

CMAKE\_MODULE\_LINKER\_FLAGS

CMAKE\_MODULE\_LINKER\_FLAGS\_DEBU

CMAKE\_MODULE\_LINKER\_FLAGS\_MINS

CMAKE\_MODULE\_LINKER\_FLAGS\_RELE

CMAKE\_MODULE\_LINKER\_FLAGS\_RELW

CMAKE\_NM /usr/bin/nm

CMAKE\_OBJCOPY /usr/bin/objcopy

CMAKE\_OBJDUMP /usr/bin/objdump

CMAKE\_RANLIB /usr/bin/ranlib

CMAKE\_SHARED\_LINKER\_FLAGS

CMAKE\_SHARED\_LINKER\_FLAGS\_DEBU

CMAKE\_SHARED\_LINKER\_FLAGS\_MINS

CMAKE\_SHARED\_LINKER\_FLAGS\_RELE

CMAKE\_SHARED\_LINKER\_FLAGS\_RELW

CMAKE\_SKIP\_RPATH OFF

CMAKE\_STRIP /usr/bin/strip

CMAKE\_USE\_RELATIVE\_PATHS OFF

CMAKE\_VERBOSE\_MAKEFILE OFF

SimTKmolmodel /bubo/home/h17/samuelf/SimTK

SimTKsimbody /bubo/home/h17/samuelf/SimTK

Now compile ..

[samuelf@tintin1 build]$ pwd

/bubo/home/h17/samuelf/svn/rnatoobox/trunk/build

[samuelf@tintin1 build]$ make -j8

# Checking out breeder with subversion (svn)

To install , do this:

mkdir ~/svn/breeder

svn checkout https://simtk.org/svn/breeder   ~/svn/breeder

If you are going to make changes to a file, it is good to make sure you have an up to date distribution:

cd ~/svn/breeder

svn update .

Let’s say you modify Breed.cpp. The way to commit this to the repository is this:

svn commit ~/svn/breeder/src/Breed.cpp

If you get an error saying the editor has not been set, this means you have to issue

export SVN\_EDITOR=vim

You should add this line to your ~/.bash\_profile so you don’t do this every time.

Reference:

http://www.techchorus.net/how-set-svneditor-environment-variable-vim

# Compiling breeder

Your LD\_LIBRARY\_PATH should point to wherever you have libMMBlib.so, e.g. :

export LD\_LIBRARY\_PATH=~/svn/RNAToolbox/trunk/build

You will then need to compile:

cd ~/svn/breeder/build

ccmake ..

Then check your cmake variables. The big pitfalls are the ones that locate your linked libraries (mysql, SimTK, MMB).

Now compile ..

[samuelf@tintin1 build]$ pwd

/bubo/home/h17/samuelf/svn/breeder/build

[samuelf@tintin1 build]$ make -j8

Linking CXX executable breeder

[100%] Built target breeder

[samuelf@tintin1 build]$

# Running

To run, use:

~/svn/breeder/build/breeder -ID 14MAY13 -MLF ./mutations-list.txt

Where the file specified by –MLF contains as many mutants as you wish, in the format : X-NNI-S.x-nni-s where X is the chain ID, NNI is the residue ID and insertion code (if any) and S is the substituted residue type. Separate the mutants with “.” For example:

C-128-V.C-147-H.C-163-N

C-128-R.C-147-H.C-163-N

Breeder will fetch all run parameters (except the actual mutations) from the mysql jobInfo table:

mysql> select \* from jobInfo;

+--------------------------------------------------------+--------------------------------------------------------------+------------+----------------+----------------+---------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------------+-----------------------+-----------------------------------------------------------+

| runID | batch\_directory | maxMutants | generationSize | maxGenerations | decayConstantsBetweenHighestAndLowest | wildTypeSequence | reportingInterval | numReportingIntervals | wildTypeStructureFileName |

+--------------------------------------------------------+--------------------------------------------------------------+------------+----------------+----------------+---------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------------+-----------------------+-----------------------------------------------------------+

| 14MAY13 | /bubo/home/h17/samuelf/glob/projects/protein-A/14MAY13 | 4 | 2 | 1000 | 5 | C:FNKEQQNAFYEILHLPNLNEEQRNGFIQSLKDDPSQSANLLAEA.D:PSVFLFPPKPKDTLMISRTPEVTCVVVDVSHEDPQVKFNWYVDGVQVHNAKTKPREQQYNSTYRVVSVLTVLHQNWLDGKEYKCKVSNKALPAPIEKTISKAKGQPREPQVYTLPPSREEMTKNQVSLTCLVKGFYPSDIAVEWESNGQPENNYKTTPPVLDSDGSFFLYSKLTVDKSRWQQGNVFSCSVMHEALHNHYTQKSLSLS | 1 | 15 | /bubo/home/h17/samuelf/glob/projects/protein-A/last.1.pdb |

+--------------------------------------------------------+--------------------------------------------------------------+------------+----------------+----------------+---------------------------------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------------+-----------------------+-----------------------------------------------------------+

7 rows in set (0.00 sec)

mysql>

You can change reportingInterval and numReportingIntervals on jobInfo, but as yet not other MMB parameters. If you wish to modify MMB’s other parameters, to go into commands.dat, modify src/Chromosome.cpp method writeCommonPart:

void Chromosome::writeCommonPart(std::ofstream & output, MysqlConnection & mysqlConnection) {

output <<"firstStage 2 "<<std::endl;

output <<"lastStage 2"<<std::endl;

output <<"loadSequencesFromPdb "<<std::endl;

output <<"smallGroupInertiaMultiplier 11 "<<std::endl;

output <<"#reportingInterval .000000000000000000000001"<<std::endl;

output <<"reportingInterval ";

output <<mysqlConnection.getReportingInterval();

output <<std::endl;

output <<"numReportingIntervals "; //150 "<<std::endl;

output << mysqlConnection.getNumReportingIntervals();

output <<std::endl;

output <<"#numReportingIntervals 1 "<<std::endl;

output <<"readAtStage 2"<<std::endl;

output <<"globalBondTorsionScaleFactor 1"<<std::endl;

output <<"globalBondBendScaleFactor 1"<<std::endl;

output <<"globalCoulombScaleFactor 1"<<std::endl;

output <<"globalVdwScaleFactor 1"<<std::endl;

output <<"globalAmberImproperTorsionScaleFactor 1"<<std::endl;

output <<"readBlockEnd"<<std::endl;

output <<"randomizeInitialVelocities false"<<std::endl;

output <<"mobilizer Rigid"<<std::endl;

output <<"constrainChainRigidSegments"<<std::endl;

};

You will then need to recompile:

[samuelf@tintin1 build]$ pwd

/bubo/home/h17/samuelf/svn/breeder/build

[samuelf@tintin1 build]$ make -j8

Linking CXX executable breeder

[100%] Built target breeder

[samuelf@tintin1 build]$

## Starting a new job

First, go to the jobInfo table and specify the parameters for the job, e.g.:

insert into jobInfo (runID , batch\_directory , ,reportingInterval ,numReportingIntervals , wildTypeStructureFileName) VALUES (“17MAY13” , “/bubo/home/h17/samuelf/glob/projects/recA/17MAY13” , 0.1, 150, “/bubo/home/h17/samuelf/glob/projects/recA/last.1.pdb”);

Now put all your mutants in a text file. Name whatever you want (e.g. myMutants.txt). Then run as:

./breeder -ID 17MAY13 -MLF ./myMutants.txt

## Useful MySQL commands:

Include update and delete

update jobInfo set lastStage = 3 where runID = “14MAY13”;

Be very careful with the delete command! Always run it with a carefully selected “where” restriction:

delete from jobInfo where runID = “14MAY13”;