Quick-start guide to protein structure clustering using Rosetta 3.1

This is one of a series of tutorials designed to get you started with using Rosetta 3.1. It was produced to accompany Kaufmann et. al. (2010) Biochemistry, and the latest version can be found at http://meilerlab.org/. Rosetta typically produces thousands of structures in the course of its output, and selecting the best result would naively mean just choosing the one with the lowest energy (score). However, the energy landscape of protein folding is extremely rugged, and in the absence of perfect sampling, there is no guarantee that the global minimum has been found. Therefore it is frequently useful to identify the largest cluster of decoys, with the idea that the broadest energy well is often also the deepest. An intelligent combination of clustering and energy ranking can help identify the most native-like structure, and Rosetta 3 is distributed with basic clustering functionality included.

It is assumed that you have installed the Rosetta suite from rosettacommons.org, and that you are comfortable working in Linux. You will also want to become familiar with the documentation that can be found in the /manual/ and /demos/ subdirectories, as well as the online Rosetta 3 User Manual, FAQ, and forums at http://www.rosettacommons.org/tiki/.

Input for the clustering run

1. You need a set of decoys, or output structures, to cluster. These can be in individual pdbfiles, or combined in the silentfile format. There is a sample silent.out file provided with this tutorial in the /cluster/ directory included with this tutorial; it contains 50 structures generated by folding the ubiquitin sequence.
2. Create a flags file to specify the parameters Rosetta will use during this run, including the location and format of input and output files, and options that specify details of the algorithm. There is a sample flags file in the /cluster/ directory. It is worth reading the file in a text editor becuase it is well-commented and contains pointers to more documentation. The parameters of the run can be changed by editing this file.
3. Conduct the clustering run:

```
cluster.linuxgccrelease @flags >& cluster.log
```

Analysis

1. Examine the output files generated during this clustering run. This includes the logfile, the pdb structure files, and the scorefile. These are all plain text files, so you can read them with a pager or text editor. clustered. The logfile has a summary of the results at the end. The output structures are given names in the form of c.i.j which denotes the jth member of the ith cluster (starting from zero).
2. Rosetta's output structures are often referred to as decoys. Load some of the decoys into your favorite molecular visualization program (Chimera, PyMOL, etc) to verify that the clusters look reasonable to you. By adjusting parameters in the flags file, you can set the clustering to generate anything from one cluster per input file, to one cluster containing all the input files. You will need to customize the parameters according to your input dataset. It can be worthwhile to use the -nooutput flag while doing that.