Domain-level Reaction Enumerator

Karthik Sarma, Casey Grun, and Erik Winfree.

Overview

This package predicts the set of possible reactions between a set of initial nucleic acid complexes. Complexes are comprised of strands, which are subdivided into "domains"—contiguous regions of nucleotide bases which participate in Watson-Crick hybridization. The enumerator only considers reactions between complexes with complementary domains. At this point, only unpseudoknotted intermediate complexes are considered.

This package is written for Python 2.7; Python must be installed and in the user's `path` in order to run the program.

Usage

```
usage: enumerator.py [-h] [--infile INPUT_FILENAME]
                     [--outfile OUTPUT_FILENAME] [-o OUTPUT_FORMAT]
                     [-i INPUT_FORMAT] [-c]
                     [--max-complex-size MAX_COMPLEX_SIZE]
                     [--max-complexes MAX_COMPLEX_COUNT]
                     [--max-reactions MAX_REACTION_COUNT]

optional arguments:
  -h, --help            show this help message and exit
  --infile INPUT_FILENAME
                        Path to the input file
  --outfile OUTPUT_FILENAME
                        Path to the output file
  -o OUTPUT_FORMAT      Desired format for the output file
  -i INPUT_FORMAT       Desired format for the input file
  -c                    Condense reactions into only resting complexes
  --max-complex-size MAX_COMPLEX_SIZE
                        Maximum number of strands allowed in a complex (used
                        to prevent polymerization)
  --max-complexes MAX_COMPLEX_COUNT
                        Maximum number of complexes that may be enumerated
```

```
                      before the enumerator halts.
  --max-reactions MAX_REACTION_COUNT
                      Maximum number of reactions that may be enumerated
                      before the enumerator halts.
```

Building documentation

Documentation is built from comments in the source using Sphinx; Sphinx must be installed. Then you can run:

```
make docs
```

from within the main directory to build HTML documentation. Additional output formats are available, and can be generated by moving to the `docs/` subdirectory and using `make`. Type `make` to show a list of available output formats.

Running unit tests

Unit tests for the project are written using Nosetests. Nosetests must be installed. Then you can run:

```
make tests
```

from within the main directory to run unit tests.