Kiwi Blast/MySQL Database Tool

Version 0.3

Shawn.M.Rupp@asu.edu

Contents

Introduction

Kiwi is a series of scripts for uploading NCBI Viral RefSeqs from a flat file to a MySQL database, making DNA and protein Blast databases, running blastz and blastn, and concatenating the results with additional information from the MySQL database.

Dependencies:
Python3
MySQL
Cython
NCBI Blast+ 2.2.31+
usearch v10.0.240

Installation

MySQL
To install MySQL on any Debian-based Linux platform, enter the following into a terminal:

```
sudo apt-get install mysql-server
sudo apt-get install mysql-client
```

You only need the client for Kiwi, but in many other cases the server is required.

Cython
Several of Kiwi's scripts utilize Cython, which compiles python code into C and drastically improves performance. Cython can be installed from the pypi repository or via Miniconda (it is installed by default with the full Anaconda package).

To install with Miniconda:
```
conda install cython
```

NCBI Blast+
Downloads and user manuals can be found here. Be sure to add blast to your path before running Kiwi.

Usearch
If you wish to use ublast(which is much faster than blast) you must download the appropriate binary for you system. After downloading, change the name of the binary to just "usearch" and move it into Kiwi/bin/.

Kiwi
Download the repository:

```
git clone https://github.com/icwells/Kiwi.git
```

Most of the scripts are written in python3, but several contain Cython modules which must be compiled. Running the install script will compile the Cython scripts and move the binaries to the bin directory:

Change into the Kiwi head directory and run the install script:

```
cd Kiwi/
./install.sh
```

Getting Started

Downloading Flat Files and Initializing Database
RefSeqs for viruses can be downloaded here (only viral.1.genomic.gbff.gz anf viral.2.genomic.gbff.gz are needed). These files can be concatenated into one or uploaded separately. Before uploading, be sure to create a MySQL database named "ASUviralDB" with a table called "Annotations" with the columns and types from tableColumns.txt (located in Kiwi/bin/). You may make another table for proteins, or initialize later with the "--new" flag.

Uploading Flat Files to Database
After downloading the RefSeq flat files and initializing a MySQL database with Annotations table, you can upload the flat files to the database:

```
cd bin/
python uploadFlatFile.py –u <MySQL username> –t <protein table name>
     -i <path to flat file> <--new>
```

If you are uploading to a protein table that has not been made yet, you may pass the –new flag, which will create a new table with the name given by -t. After hitting enter, you will be prompted for your MySQL pasword. This script will then upload the appropriate data to the Annotations table and the given protein table.

Extracting Data from the Database

```
python extractData.py –u <MySQL username> –t <protein table name>
     -d <path to output directory> --<dna/protein/csv> -p <# of
     threads>
```

This script follows a similar pattern to uploadFlatFile. Instead of an input file, you must supply an output directory where output will be written. Output can be in csv format (with the sequences omitted since they are often too large) or fasta format for DNA or protein sequences. Fasta outputs can be extracted in parallel since the number of queries to SQL and the amount of data can make this process time consuming.

Building Blast Databases, Running Blast, and Concatenating Output

```
python blastSeqs.py --nucdb --protdb -d <database directory>
```

The above command will convert the output from the previous script to blast databases. Both of the –-nucdb –-protdb options may be used at the same time, or they can be run one at a time.

```
python blastSeqs.py -p <# of threads> -d <database directory>
     -o <output directory> -i <input fasta>
```

This command will call blastx on the input fasta file (must be DNA sequences), subset sequences with an e value less than $10^{-5,}$, and call blastn on the subset sequences. The database directory is the same as above, and all output is written to the output directory.

```
python concatenateOutput.py –u <MySQL username> –o <output directory>
     –i <input fasta>
```

This script will concatenate blastn and blastx results in the output directory with annotation information from the database. It will also include query DNA sequences, so it requires the input fasta file used in the previous step.

Updating the Databases

```
python updateDatabases.py –u <MySQL username> –d <database directory>
```

This will add any new entries to both the MySQL database and the blast databases.

Scripts

uploadFlatFile.py
This script uploads an NCBI flat file to ASUviralDB.

```
python uploadFlatFile.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  -u U        MySQL username.
  --new       Initializes a new table.
  -t T        Name of table to upload data to
  -i I        Path to input flat file.
```

extractData.py
This script will extract data from the ASUviralDB in various formats.

```
python extractData.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  --backup    Backup database to local Linux machine.
  -u U        MySQL username.
  -t T        Name of table to extract data from (not needed for
               exacting dna
              sequences).
  --csv       Extract all data from a given table to a csv (sequences
               will be omitted).
  --dna       Extract dna sequences from database in fasta format.
  --protein   Extract protein sequences from database in fasta
               format.
  --genes     Extracts genes using coordinated from given table.
  -d D        Path to output directory.
```

blastSeqs.py
This script will automate running blastx and blastn for putative viral sequences. Be sure to export the path to blast on your machine before using.

```
python blastSeqs.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  -i I        Path to fasta file of query sequences.
  -o O        Path to working/output directory.
  -d D        Path to directory containing database files.
  --blast     Runs BLAST (default is ublast).
  -p P        Number of threads to run BLAST or ublast with.
  --ublastdb  Creates new ublast DNA and protein databases. Place
               source fasta files into the directory specified with
               -d.
  --blastdb   Creates new blast DNA and protein databases. Place
               source fasta files into the directory specified with
               -d.
```

concatenateOutput.py
This script produces a csv file of significant blastx and blastn output and includes relevant information from the SQL database.

```
python concatenateOutput.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  -u U        MySQL username.
  -i I        Path to fasta file of query sequences.
  -o O        Path to input directory (output directory from
               blastSeqs).
  --blast     Indicates BLAST was run in the previous step (default
               is ublast).
```

updateDatabases.py
This script will download the current NCBI RefSeq flat files for viral DNA and add any new entries to the MySQL database. It will then append new entries to the fasta files in the database directory and create new blast databases from them.

```
python updateDatabases.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  --id        Resumes script from identification phase (skips
               download).
  --upload    Resumes script from uploading to MySQL database
               (requires subset flat file).
  --extract   Resumes script from extracting from MySQL database (new
               entries must have been uploaded).
  -u U        MySQL username.
  -d D        Path to blast database directory.
```

subsetFlatFile.py
This script will extract entries from the target flat file which failed uploading to the MySQL database. Many of these are due to runtime issues, so they may upload without a problem on the second attempt.

```
python subsetFlatFile.py
  -h, --help  show this help message and exit
  -v          Prints version info and exits.
  i I        Path to input flat file.
```