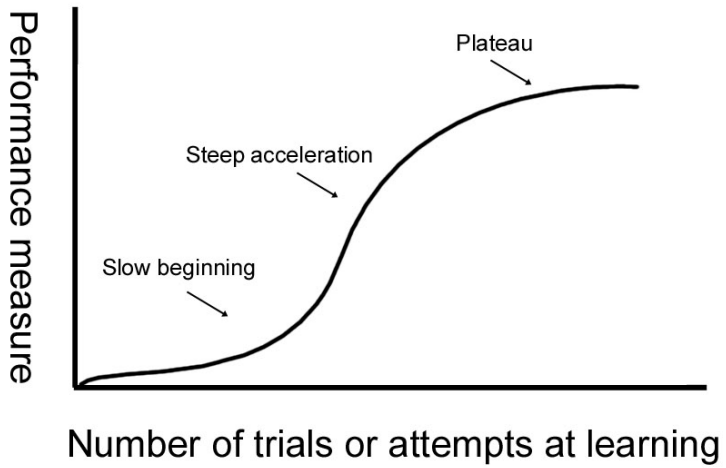
Install and run external command line softwares

Yanbin Yin

Fall 2015



Homework #8

Create a folder under your home called hw8

Change directory to hw8

Copy the faa file and ptt file you've downloaded from NCBI ftp site in hw7 to hw8 folder

Create unix one-liner (piping different commands) to count how many proteins are there in the faa file and save the protein gi numbers (just the gi) in a file

Create unix one-liner (piping different commands) to process the ptt file and count how many genes are located in the positive strand and how many in the negative strand

Write a report (in word or ppt) to include all the operations/commands and screen shots.

Due on Nov 24 (send by email)

Office hour:
Tue, Thu and Fri 2-4pm, MO325A
Or email: yyin@niu.edu

Programs/tools we often use

- BLAST
- FASTA
- HMMER
- EMBOSS
- lftp
- bioperl
- R

- Clustalw
- MAFFT
- MUSCLE
- SRA toolkit
- weblogo
- PhyML
- FastTree
- RaxML
- USEARCH
- …

http://gacrc.uga.edu/

On WINDOWS, you download some softwares usually in exe format. You double click on some icon and windows installer will do the rest for you. Usually you will be asked: where you want to have this program installed to (C:\WINDOWS\Programs\)

On Linux, it's very different. You download the software package, usually in compressed format (.gz or .zip). You have to type in a series of commands to install it, often write it to a system folder (e.g. /usr/bin), which you don't have access unless you are the root user

Linux-based program types

- Source codes in C, C++, Java, Fortran etc.
  - Need to be compiled before execute the command
- Precompiled executables or binary codes
- Source codes in scripting languages (perl, python, R etc.)
  - Can execute directly

Many programs need dependencies (if order to install program A, you need install B first …)

On your own ubuntu machine …

Not on Ser

- You are the root (administrator) and using the sudo command you can install anything you want into the system directory (/usr/bin/, /bin/, /lib/ etc.)
  - apt-get (Advanced Packaging Tool) can do many installations for you from source or binary codes

  https://help.ubuntu.com/12.04/serverguide/apt-get.html

- On Ser, you are not the root and you can only install things under your home using the "hard" way
  - Download->unpack->install->edit PATH environmental variable
  - Make sure you create folders for each tools, e.g. yourhome/tools/fasta

Install BLAST on your own machine

Not on Ser

[test if installed]

```
sudo apt-get install blast2
```

[test if installed]

```
blastall
```

[where it installed]

```
which blastall
```

[if you want to uninstall]

```
sudo apt-get remove blast2
```

[test if it's gone]

```
blastall
```

[new version of blast]

```
sudo apt-get install ncbi-blast+
```

Use apt-get to install

lftp, emboss, hmmer, bioperl, clustaw, muscle, R

```
sudo apt-get install xxx
```

[to test if installed, type in the command]

On your own MAC

http://www.digimantra.com/howto/apple-aptget-command-mac/
http://superuser.com/questions/173088/apt-get-on-mac-os-x

http://www.macobserver.com/tmo/article/install_the_command_line_c_compilers_in_os_x_lion

Install Xcode, then C compiler, then you can install Mac port

http://www.macports.org/install.php

With mac port, you can install

wget: `sudo port install wget`

lftp : `sudo port install lftp`

hmmer: `sudo port install hmmer`

emboss: `sudo port install emboss`

R: `sudo port install R`

blast:
http://www.blaststation.com/freestuff/en/howtoNCBIBlastMac.html

Install programs using the hard way (your are NOT the root)

Usually when a bioinformatics tool is released, the tool package also includes a readme, install or manual file in addition to the program itself.

In most cases, there are more than one files in the program folder.

Some times you will see files in multiple different languages.

The readme or install file will tell you how to install the tool.

Basic steps: download the package -> unpack (untar, unzip) -> read the readme -> install

Install BLAST using the hard way

BLAST source is written C and C program needs to be compiled to get executable program
But here we will download **executables** of BLAST

```
lftp ftp.ncbi.nih.gov:/blast/executables/LATEST
get ncbi-blast-2.2.31+-x64-linux.tar.gz
bye
```

Now you returned to Ser

```
ls -l
mkdir tools
cd tools
mkdir blast
mv ../ncbi-blast-2.2.31+-x64-linux.tar.gz blast
cd blast
```

Unpack the tar ball

```
tar -zxf ncbi-blast-2.2.31+-x64-linux.tar.gz
ls -l
cd ncbi-blast-2.2.31+/bin
ls -l
./blastp -h
pwd
```

Now return to your home

```
cd
blastp -h
```

You will be told that command is not found. The reason is that blastp is not in your current path. You have to give the full path to run it in your home

```
/home/yyin/tools/blast/ncbi-blast-2.2.31+/bin/blastp
```

What if you don't want to type this long path? You can add it to a hidden file in your home called .bashrc where Shell auto execute every time you log in: Shell will be notified that when you call blastp, go to that folder to find it.

You are at your home

```
nano .bashrc
```

Add the following in the beginning of the file

```
export PATH="$PATH:$HOME/tools/blast/ncbi-blast-2.2.30+/bin"
```

Now exit from nano and do the following

```
. .bashrc
blastp -h
```

Environment variable

An environment variable is a named object that contains data used by one or more applications. The value of an environmental variable can for example be the location of all executable files in the file system, the default editor that should be used, or the system locale settings. Users new to Linux may often find this way of managing settings a bit unmanageable. However, environment variables provides a simple way to share configuration settings between multiple applications and processes in Linux.

`env` to list all built-in environment variable

PATH is a very important environment variable. This sets the path that the shell would be looking at when it has to execute any program. It would search in all the directories that are defined in the variable. Remember that entries are separated by a ' : ' . You can add any number of directories to this list.

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games

Run BLAST in command line mode

http://www.ncbi.nlm.nih.gov/books/NBK52640/

Why you do need to run BLAST in command line terminal?

- NCBI's server does not have a database that you want to search
- Millions of users are using NCBI BLAST server too
- Your query set has more than one sequence or even a genome
- The BLAST output could be processed and used as input for other Linux softwares

For doing blast search

1. Determine what is your query and database, and what is the command
2. Format your database
3. Run the blast command (what E-value cutoff, what output format)
4. View the result
5. Parse the result (hit IDs, hit sequences, alignment, etc.)

There are two versions of BLAST
-blast2 (legacy blast)
-blast+ (this is what you installed in your home/tools/)

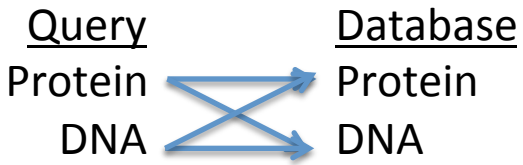
https://www.biostars.org/p/9480/
http://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=Download

BLAST2 (legacy blast)

```
blastall - | less
-p # specify blastp, blastn, blastx, tblastn, tblastx

More commands in blast package

formatdb (format database)
megablast (faster version of blastn)
rpsblast (protein seq vs. CDD PSSMs)
impala (PSSM vs protein seq)
bl2seq (two sequence blast)
blastclust (given a fasta seq file, cluster them based on sequence similarity)
blastpgp (psi-blast, iterative distant homolog search)
```



http://www.ncbi.nlm.nih.gov/books/NBK1763/pdf/ch4.pdf

blastall options

-p program name
-d database file name (text fasta sequence file)
-i query file name
-e e-value cutoff (show hits less than the cutoff)
-m output format
-o output file name (you can also use >)
-F filter low-complexity regions in query
-v number of one-line description to be shown
-b number of alignment to be shown
-a number of processers to be used

New version blast+, e.g. blastp

At your home

```
ls -l tools/ncbi-blast-2.2.31+/bin/
```

22 executable files (commands)

blastp
blastn
tblastn
blastx
tblastx

```
blastp -help | less
```

-query query file name

-db database file name

-out output file name

-evalue e-value cutoff

-outfmt output format

-num_descriptions

-num_alignments

http://www.ncbi.nlm.nih.gov/books/NBK1763/pdf/CmdLineAppsManual.pdf

Exercise 1: find CesA/Csl homologous sequence in charophytic green algal genome

Klebsormidium flaccidum genome was sequenced, analyzed and published in http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4052687/. We want to find out how many CesA/Csl genes this genome has.

Note that the raw DNA reads are available in GenBank, but the assembled genome and predicted gene models are not

Go to Methods of the above paper and find the link to the website that provides the genome and annotation data

Wget the predicted protein sequence file

```
wget -q http://www.plantmorphogenesis.bio.titech.ac.jp/~algae_genome_project/klebsormidium/kf_download/131203_kfl_initial_genesets_v1.0_AA.fasta
```

Make index files for the database

```
makeblastdb -in 131203_kfl_initial_genesets_v1.0_AA.fasta -parse_seqids -dbtype 'prot'
```

Check how many proteins have been predicted

```
less 131203_kfl_initial_genesets_v1.0_AA.fasta | grep '>' | wc
```

Wget our query set

```
wget -q http://cys.bios.niu.edu/yyin/teach/PBB/cesa-pr.fa
```

Do the blastp search

```
blastp -query cesa-pr.fa -db 131203_kfl_initial_genesets_v1.0_AA.fasta -out
cesa-pr.fa.out -outfmt 11
```

Check out the output file

```
less cesa-pr.fa.out
```

Change the output format to a tab-delimited file

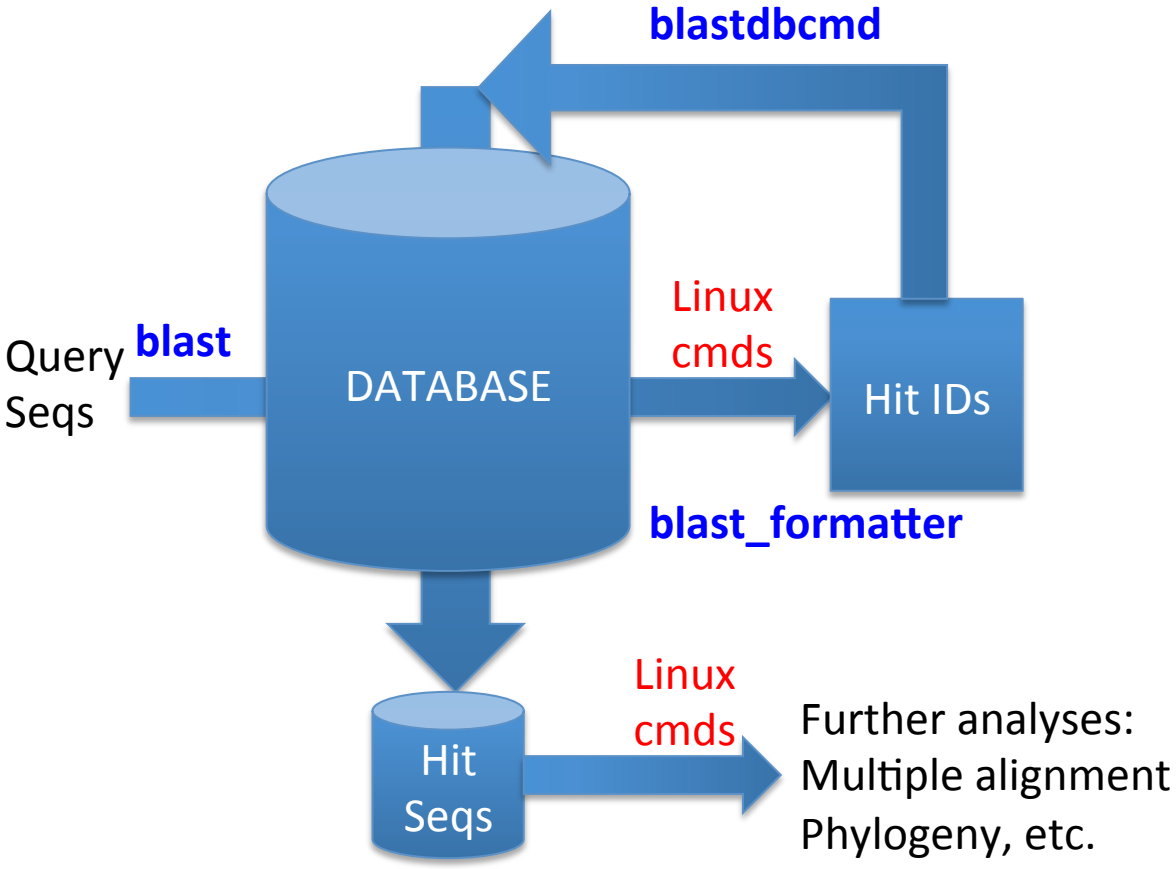
```
blast_formatter -archive cesa-pr.fa.out -outfmt 6  | less
```

Filter the result using E-value cutoff

```
blast_formatter -archive cesa-pr.fa.out -outfmt 6 | awk '$11<1e-10'  | less

blast_formatter -archive cesa-pr.fa.out -outfmt 6 | awk '$11<1e-10' | cut -f2 |
less
```

How do you extract the sequences of the blast hits?



Save the hit IDs

```
blast_formatter -archive cesa-pr.fa.out -outfmt 6 | awk '$11<1e-10' | cut -f2 |
sort -u > cesa-pr.fa.out.id
```

Retrieve the fasta sequences of the hits

```
blastdbcmd -db 131203_kfl_initial_genesets_v1.0_AA.fasta -entry_batch cesa-
pr.fa.out.id | less

blastdbcmd -db 131203_kfl_initial_genesets_v1.0_AA.fasta -entry_batch cesa-
pr.fa.out.id | sed 's/>lcl|/>/' | sed 's/ .*//' | less

blastdbcmd -db 131203_kfl_initial_genesets_v1.0_AA.fasta -entry_batch cesa-
pr.fa.out.id | sed 's/>lcl|/>/' | sed 's/ .*//’ > cesa-pr.fa.out.id.fa
```

Wget the annotation file from the Japan website

```
wget -q http://www.plantmorphogenesis.bio.titech.ac.jp/~algae_genome_project/
klebsormidium/kf_download/131203_gene_list.txt
```

Check what our genes are annotated to be

```
grep -f cesa-pr.fa.out.id 131203_gene_list.txt
```

If a program (e.g. BLAST) runs so long on a remote Linux machine that it won't finish before you leave for home …

Or if you somehow want to restart your laptop/desktop where you have a Putty session is running (Windows) or a shell terminal is running (Ubuntu) …

In any case, you have to close the terminal session (or have it be automatically terminated by the server). If this happens, your program will be terminated without finishing. If you expect your program will run for a very long time, e.g. longer than 10 hours, you may put "nohup" before your command; this ensures that even if you close the terminal, the program will still run in the background until it is finished and you can log in again the next day to check the output. For example:

```
nohup blastp -query yeast.aa -db yeast.aa -out yeast.aa.ava.out
-outfmt 6 &
```

You will get an additional file nohup.out in the working folder and this file will be empty if nothing wrong happened.

Exercise 2: emboss

You are at your home

```
cd tools
lftp emboss.open-bio.org
```

Inside emboss lftp site

```
cd pub/EMBOSS
get emboss-latest.tar.gz
```

Bye to exit from lftp

```
bye
```

Now you are back to your home on Ser

```
tar zxf emboss-latest.tar.gz

./configure –prefix=$HOME/tools/emboss

make

make install
```

If you are the root, one command can do all the above for you

```
sudo apt-get install emboss
```

Test if emboss has been installed:

```
water
```

Change sequence format

```
seqret -help
```

```
seqret -sequence /home/yyin/Unix_and_Perl_course/Data/GenBank/
E.coli.genbank -outseq E.coli.genbank.fa -sformat genbank -osformat fasta
```

Calculate sequence length

```
infoseq -help
```

```
infoseq -sequence /home/yyin/cesa-pr.fa -name -length -only
```

More command examples:

```
needle -help
water -help
fuzznuc -help
pepstats -help
pepinfo -help
plotorf  -help
transeq  -help
prettyseq -help
```