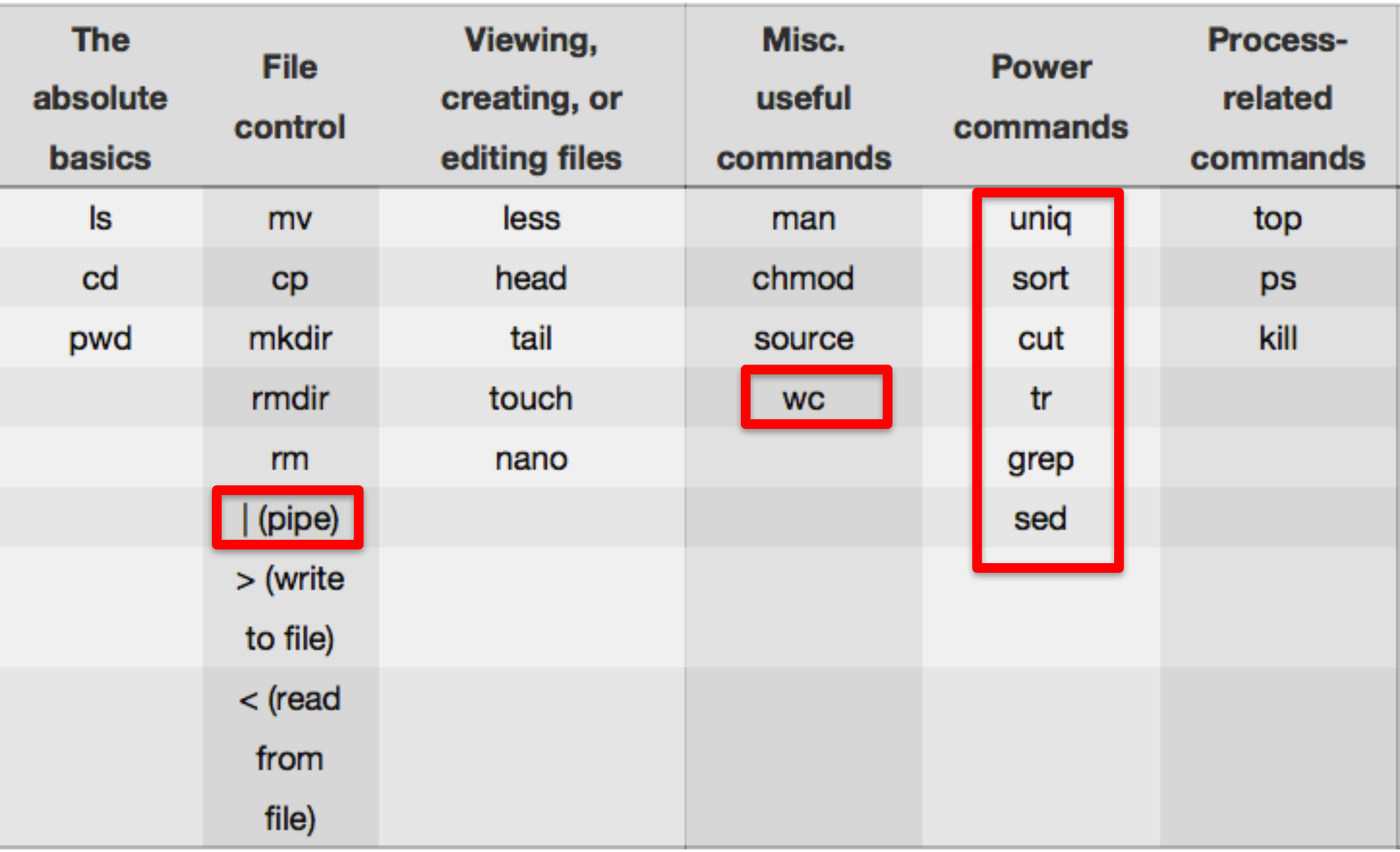
Linux command line basics III: piping commands for text processing

Yanbin Yin



The absolute basics	File control	Viewing, creating, or editing files	Misc. useful commands	Power commands	Process-related commands
ls	mv	less	man	uniq	top
cd	cp	head	chmod	sort	ps
pwd	mkdir	tail	source	cut	kill
	rmdir	touch	wc	tr	
	rm	nano		grep	
	| (pipe)			sed	
	> (write to file)				
	< (read from file)				

http://korflab.ucdavis.edu/Unix_and_Perl/unix_and_perl_v3.1.1.pdf

The beauty of Unix for bioinformatics
sort, cut, uniq, join, paste, sed, grep, awk, wc, diff, comm, cat

All types of bioinformatics sequence analyses are essentially text processing.

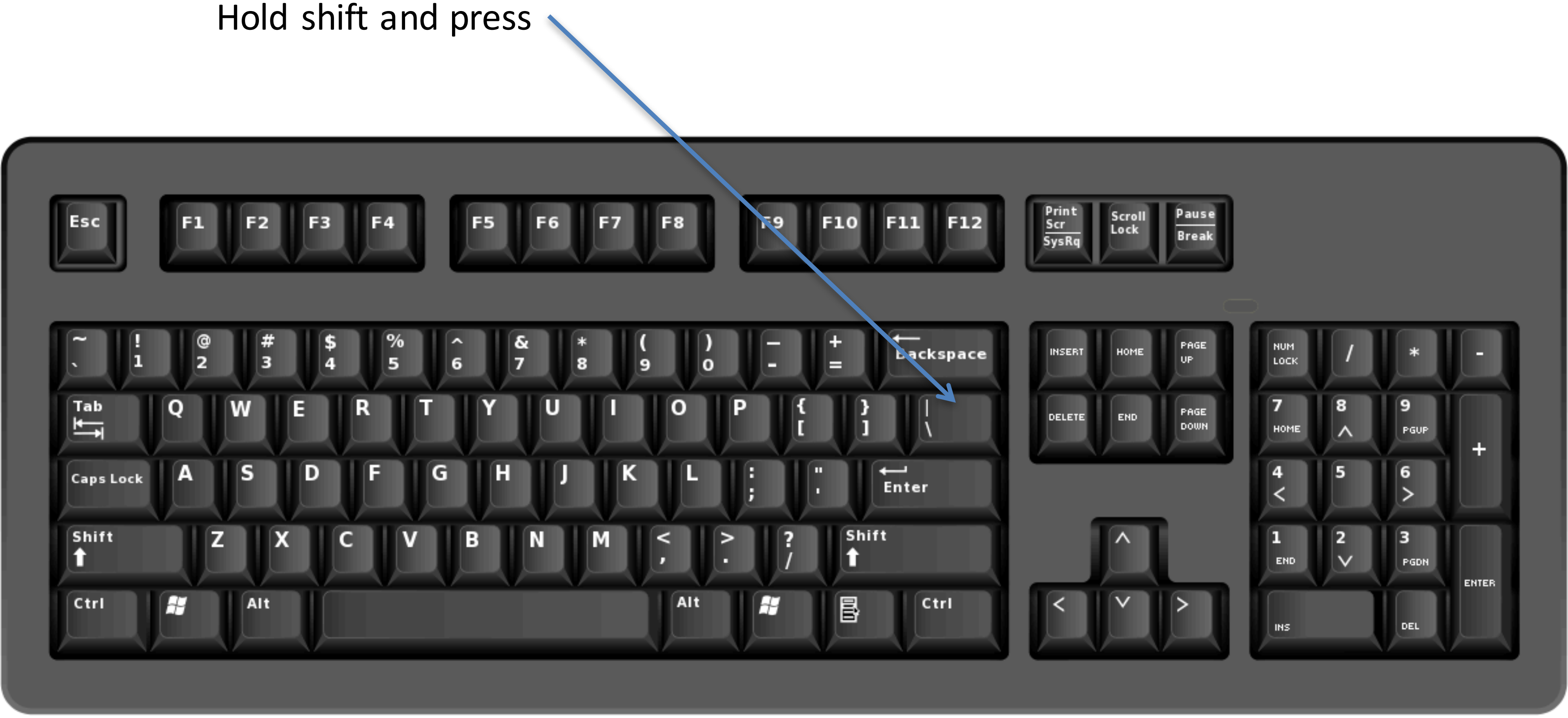
Unix Shell has the above commands that are very useful for processing texts and also allows **the output from one command to be passed to another command as input using pipes** (“|”).

This makes the processing of files using Shell very convenient and very powerful: you do not need to write output to intermediate files or load all data into the memory.

For example, combining different Unix commands for text processing is like passing an item through a manufacturing pipeline when you only care about the final product

|

Hold shift and press


Esc
F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12
Print Scr SysRq
Scroll Lock
Pause Break
~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Alt Alt Ctrl
INSERT HOME PAGE UP
DELETE END PAGE DOWN
NUM LOCK / * -
7 HOME 8 9 PGUP +
4 5 6
1 END 2 3 PGDN ENTER
INS DEL

cut: extract columns from a file

```
less file | cut -f1 # cut the first column (default delimiter tabular key)
less file | cut -f1 -d ' '  # specify delimiter to be regular space
less file | cut -f1-3     # cut 1 to 3 col
less file | cut -f1,7,10 > file.1-7-10     # cut 1, 7, 10 col and save as a new file
```

sort: sort rows in a file, default on first col in alphabetical order (0-9 then a-z, 10 comes before 9)

```
less file | sort -k 2  # sort on 2 col
less file | sort -k 2,2n # sort in numeric order
less file | sort -k 2,2nr     # sort in reverse numeric order
```

uniq: report file without repeated occurrences

```
less file | cut -f2 | sort | uniq  # unique text
less file | cut -f2 | sort | uniq -c    # count number of occurrences of unique texts
```

grep: extract lines match a given word or pattern

```
less file | grep '>' | head  # print only lines containing '>'
less file | grep -v '>' | head # print lines not containing '>'
less file | grep -n '>' | head  # also print in which lines '> is found
less file | grep -c '>' # count the number of occurrences
less file | egrep 'chr1|chr2'# print lines containing chr1 or chr2 (multi-words or patterns)
```

sed: stream editor, modify, delete, search and replace etc

```
less file | grep '>' | sed 's/>//'         # delete '>'
less file | grep '>' | sed 's/>/+/'        # replace '>' with '+'
less file | sed '/^$/d'        # delete empty line
less file | sed '/>/d'        # delete all lines with '>'
less file | sed -n '/>/p'      # print all lines with '>'
less file | sed -n '101,200p'      # print selected lines (101 to 200) in the file
```

awk: give a condition, perform an action (print)

```
less file | awk '$5=="22"'  #  $5 means the 5th col, default delimiter is regular space
less file | awk -F "\t" '$5=="22"'       # define delimiter to be tabular space "\t"
less file | awk '/>/'  # put pattern between //
less file | awk '$1~/>/'            # specify the pattern appears in the 1st col
less file | awk '{print $1,$3}'         # print the 1 and 3 cols, regular space separated
less file | awk '{print $1,"new",$3}'         # insert a new col with text "new"
less file | awk '{print $3,$1}'          # change the order of 1st and 3rd col
```

Example 1: process cesa-pr fasta sequence file to get protein IDs

Remove the file if you've ever downloaded it before

```
rm cesa-pr.fa
```

Check if it is removed

```
ls -l
```

Create the file

```
nano cesa-pr.fa
```

Open this link in web browser, copy the content, go to the terminal, right click to paste in nano

View the file

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

Only keep the description line

```
less cesa-pr.fa | grep '>'
```

Get ride of '>'

```
less cesa-pr.fa | grep '>'  | sed 's/>//'
```

Replace '|' with a tab space

```
less cesa-pr.fa | grep '>'  | sed 's/>//' | sed 's/|/\t/g'
```

Cut out the 2nd and 3rd cols

```
less cesa-pr.fa | grep '>'  | sed 's/>//' | sed 's/|/\t/g' | cut -f2-3
```

grep: print lines matching a pattern

Try `man grep`

```
less cesa-pr.fa | grep '>'
```

The pattern is to get any line contains a '>'

is the same as

```
grep '>' cesa-pr.fa
```

sed: stream editor for filtering and transforming text

```
sed 's/>//'
```

The pattern is to replace > with nothing
=
delete >

```
sed 's/|/\t/g'
```

The pattern is to replace | with tab space (\t is the regex)

cut: remove sections from each line of files

```
cut -f2-3
```

-f is the option: field
2-3 could also be written as 2,3

Example 2: processing Arabidopsis CAZyme list

Go to http://www.cazy.org/e1.html, select the protein list table, Ctrl+c to copy

Create a file using vi

```
vi arabi.list
```

Hit i to change to edit mode
Right click to paste
Hit Esc to change to command mode
Shift+colon, then hit x, then hit enter

View the file

```
less arabi.list
```

Count how many proteins   wc is for counting

```
wc arabi.list
less arabi.list | wc
```

Only print the 3rd col

```
less arabi.list | cut -f3 | less
```

Only print the 2nd col

```
less arabi.list | cut -f2 | less
```

Print proteins having multi-domains

```
less arabi.list | cut -f2,3 | grep ',' | less
```

Use awk to match 2nd col to be GT2

```
less arabi.list | awk -F"\t" '$2=="GT2"' | less
less arabi.list | awk -F"\t" '$2=="GT2"' | cut -f1
```

awk: pattern scanning and text processing language

awk 'condition {action}'

Use perl one-liner to retrieve AGI number

```
less arabi.list | awk -F"\t" '$2=="GT2"' | cut -f1 | perl -n -e '/(A[t,T]\w+)/ and print $1,"\n"'
```

$/\underbrace{\texttt{(A[t,T]\textbackslash w+)}}_{\text{regex}}/$

$1 is what's matched in ()

\n: new line

Many useful one-liners:
http://genomics-array.blogspot.com/2010/11/some-unixperl-oneliners-for.html

a list of commonly used wildcards and patterns:

*	any numbers of letters, numbers and characters except for spaces and special characters, e.g. ()[]+\/$@#%;,?
.	any single letter, number and character including special characters
^	start of a line caret
$	end of a line
^$	an empty line, i.e. nothing between ^ and $
[]	create your own pattern, e.g. [ATGC] matches one of the four letters only, [ATGC]{2} matches two such letters; [0-9]: any numbers
\w	any letter (a-z and A-Z)
\d	any number (0-9)
+	previous items at least one times, e.g. \w+ matches words of any sizes
{n}	previous items n times, e.g. \w{5} matches words with exactly five letters
\s	space
\t	tabular space
\n	new line

Curly brackets

http://www.bsd.org/regexintro.html

Example 3: count how many proteins in each family

Only print the 2nd col

```
less arabi.list | cut -f2 | less
```

Sort the 2nd col in alphabetical order

```
less arabi.list | cut -f2 | sort | less
```

Only show unique lines, get ride of duplicates

```
less arabi.list | cut -f2 | sort | uniq | less
```

Show unique lines and also count the occurrences of duplicates

```
less arabi.list | cut -f2 | sort | uniq -c | less
```

Sort in reverse numerical order

```
less arabi.list | cut -f2 | sort | uniq -c | sort -nr | less
```

Example 4: find out which is the most studied human gene

Connect to NCBI ftp site

```
lftp ftp.ncbi.nih.gov
```

You're in there:

```
cd gene
cd DATA

ls

get gene2pubmed.gz

bye
```

You just returned to Ser: uncompress the file

```
gzip -d gene2pubmed.gz
```

A good way to understand a long command line:

Run each step and less to see what happened and then add the next step and less

$$\underbrace{\texttt{less gene2pubmed}}_{1}\ |\ \underbrace{\texttt{awk '\$1==9606'}}_{2}\ |\ \underbrace{\texttt{cut -f2}}_{3}\ |\ \underbrace{\texttt{sort}}_{4}\ |\ \underbrace{\texttt{uniq -c}}_{5}\ |\ \underbrace{\texttt{sort -k 1,1nr}}_{6}\ |\ \texttt{less}$$

```
less gene2pubmed

less gene2pubmed | awk '$1==9606' | less

less gene2pubmed | awk '$1==9606' | cut -f2 | less
less gene2pubmed | awk '$1==9606' | cut -f2 | sort | less

less gene2pubmed | awk '$1==9606' | cut -f2 | sort | uniq -c | less

less gene2pubmed | awk '$1==9606' | cut -f2 | sort | uniq -c | sort -k 1,1nr | head -5
```

Example 5: cosmic mutation data

We will process a tab-separated file at UCSC genome browser website:
http://genome.ucsc.edu/

Copy the link by selecting and Ctrl+C:

```
wget http://hgdownload.soe.ucsc.edu/goldenPath/hg19/database/cosmicRaw.txt.gz
ls -l
gzip -d cosmicRaw.txt.gz
ls -l
less cosmicRaw.txt
```

What are each col?
http://hgdownload.soe.ucsc.edu/goldenPath/hg19/database/cosmicRaw.sql

http://rous.mit.edu/index.php/Unix_commands_applied_to_bioinformatics

```
awk 'condition {action}'
```

```
less cosmicRaw.txt | cut -f2 | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | less


less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f1 | sort -u | wc
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | awk '$6=="liver"`
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | less
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c
less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c | sort -k 1,1nr

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | cut -f5 | sort | uniq -c | sort -k
2,2n

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f6 | sort | uniq
-c | sort -k 1,1nr

less cosmicRaw.txt | cut -f2,3,4,5,8,13 | awk '$5==22' | cut -f2 | sort | uniq
-c | sort -k 1,1nr | less
```

Finding needle in a hay stack

Find under your current folder the files ending with gff

```
find . -name "*gff”

find . -name ”*faa”

find . -name "*pdf"
```

- Save history of your commands:

```
history | less
history > hist1
```

- Send message to other online users

`write` username (ctrl+c to exit)

- Change your password

```
passwd
```

Ctrl+c to tell Shell to stop current process
Ctrl+z to suspend
bg to send to background
Ctrl+d to exit the terminal (logout)

job monitor and control

top: similar to windows task manager (space to refresh, q to exit)

w: who is there

ps: all running processes, PID, status, type
ps -ef

jobs: list running and suspended processes

kill: kill processes
kill pid (could find out using top or ps)

bg: move current process to background

fg: move current process to foreground