Unix II and Bedtools

Bioinformatics Applications (PLPTH813)

Sanzhen Liu

2/28/2017

Outline

- more Unix apps: sort, find, awk, sed, wget
- data transfer
- bedtools

sort - sort lines of text files

cat fruit.txt

```
orange  8
apple   6
peach   12
banana  5
```

sort fruit.txt

```
apple   6
banana  5
orange  8
peach   12
```

sort -k 2 fruit.txt

```
peach   12
banana  5
apple   6
orange  8
```

sort -k 2n fruit.txt

```
banana  5
apple   6
orange  8
peach   12
```

sort -k 2nr fruit.txt

```
peach   12
orange  8
apple   6
banana  5
```

sort -k 1,2 fruit.txt

```
apple   6
banana  5
orange  8
peach   12
```

http://www.theunixschool.com/2012/08/linux-sort-command-examples.html

find - search for files in a directory hierarchy

```
find [pathnames] [conditions]

# Finding files >10M
find . -size +10M

# Finding files <10M
find . -size -10M

# find a file
find -name "fruit.txt"

# find a file in the current directory
find -maxdepth 1 -name "fruit.txt"
```

find - II

```
# find files containing a specific word in its name
find -name "fruit*"

# find files whose name are not "fruit.txt"
find -not -name "fruit.txt"

# find files modified within 30 minutes
find . -mmin -30

# find files modified within 1 day
find . -mtime -1

# find files accessed within 1 hour.
find . -amin -60
```

http://www.folkstalk.com/2011/12/101-examples-of-using-find-command-in.html

awk - I

- **awk** : a programming language designed for text processing and typically used as a data extraction and reporting tool.

```
cat fruit.txt
orange   8
apple 6
peach 12
banana   5

awk '{print $1}' fruit.txt # output first field

orange
apple
peach
banana
```

awk - II

```
awk 'BEGIN {start_action} {action} END {stop_action}' filename

# add up values in a column
awk 'BEGIN {sum=0} {sum=sum+$2} END {print sum}' fruit.txt
31

# print lines that satisfying certain conditions
awk '{if ($2 > 10) print }' fruit.txt
peach  12
```

awk - III:

```
# NF - Number of fileds variable:
awk '{print NF}' fruit.txt
2
2
2
2
# NR - number of records variable:
awk 'END {print NR}' fruit.txt
4

# length of strings
awk '{print length($1)}' fruit.txt

tolower(string)
toupper(string)
```

sed - a stream editor used for modifying files in unix

fruit.txt

```
orange  8
apple   6
peach   12
banana  5
```

```
sed 's/apple/strawberry/' fruit.txt
orange 8
strawberry    6
peach  12
banana 5

sed 's/apple/strawberry/g' fruit.txt
orange 8
strawberry    6
peach  12
banana 5
```

sed - II

fruit.txt

```
orange  8
apple   6
peach   12
banana  5
```

```
sed 's/apple/{&}/' fruit.txt
orange 8
{apple}   6
peach  12
banana 5
```

```
sed '/12/ s/peach/kiwi/' fruit.txt
orange 8
apple  6
kiwi   12
banana 5
```

wget

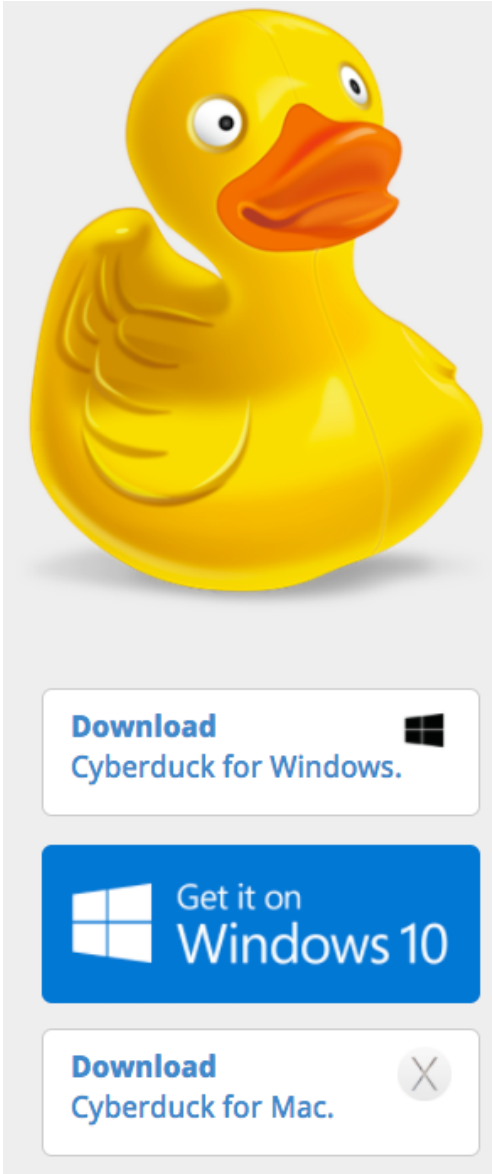
```
wget <url link to a file>
wget <a ftp link>

example:

wget http://129.130.89.83/tmp/public/sequence.cost.png
```

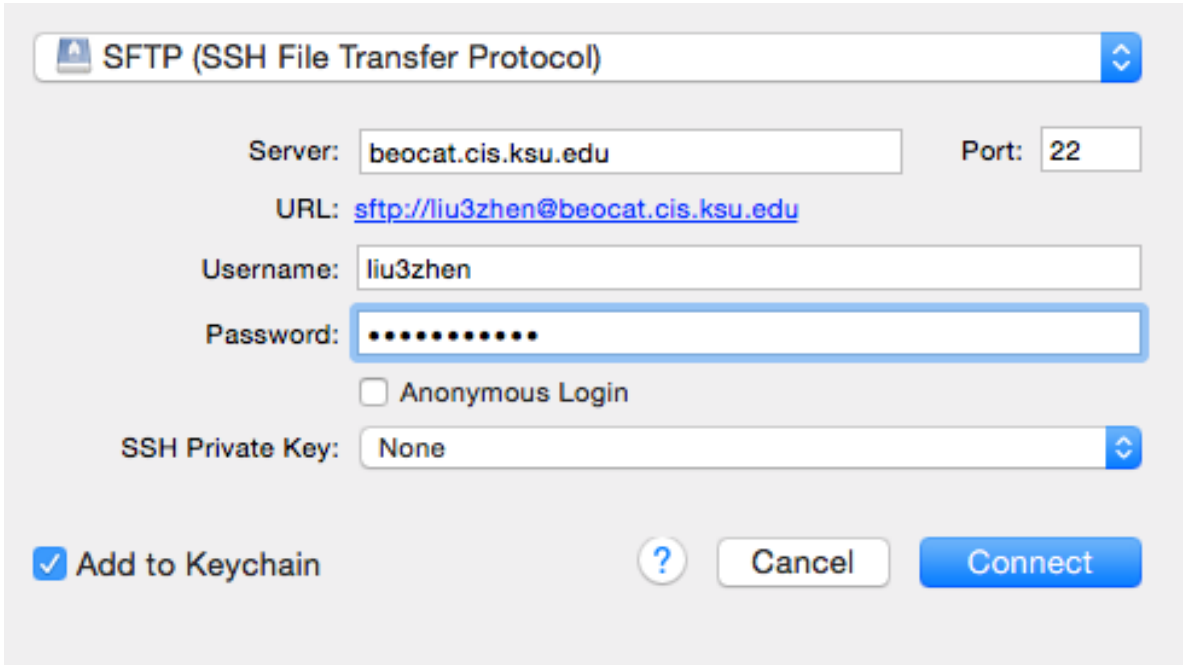
scp

```
scp user@hostname:directory/remotefile localfile
```

scp <eid>@beocat.cis.ksu.edu:<path/files> .

Cyberduck


Download
Cyberduck for Windows.
Get it on
Windows 10
Download
Cyberduck for Mac.


SFTP (SSH File Transfer Protocol)
Server:
beocat.cis.ksu.edu
Port:
22
URL:
sftp://liu3zhen@beocat.cis.ksu.edu
Username:
liu3zhen
Password:
Anonymous Login
SSH Private Key:
None
Add to Keychain
Cancel
Connect

Outline

- more Unix apps: sort, find, awk, sed, wget
- data transfer
- bedtools

BED format

The first three required BED fields are:

chrom - the chromosome
chromStart - the starting position of the feature; The first base is numbered 0.
chromEnd - the ending position of the feature in the chromosome or scaffold.
The chromEnd base is not included in the display of the feature.
For example, the first 100 bases of a chromosome are defined as
chromStart=0, chromEnd=100, and span the bases numbered 0-99.

The additional optional BED fields are:
name - Defines the name of the BED line.
score - A score between 0 and 1000
strand - Defines the strand - either '+' or '-'.
…

VCF format

```
##fileformat=VCFv4.2
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=file:///seq/references/1000GenomesPilot-NCBI36.fasta
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sap
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF    ALT     QUAL FILTER INFO                              FORMAT      N
20     14370   rs6054257 G      A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2           GT:GQ:DP:HQ C
20     17330   .         T      A       3    q10    NS=3;DP=11;AF=0.017               GT:GQ:DP:HQ C
20     1110696 rs6040355 A      G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1
20     1230237 .         T      .       47   PASS   NS=3;DP=13;AA=T                  GT:GQ:DP:HQ C
20     1234567 microsat1 GTC    G,GTCT  50   PASS   NS=3;DP=9;AA=G                    GT:GQ:DP    C
```

coverage

bedtools coverage -abam $bam -b $bed

Bed input: Interval_1 1 16452 10 -3.84

…

Output:

					1	2	3	4
Interval_1	1	16452	10	-3.84	**5432**	**16302**	**16451**	**0.9909**

1. Read number
2. Coverage (bp)
3. Interval length
4. Coverage (%)

annotate and intersect

Input 1:	#Chr	chrStart	chrEnd	Len
	10	10006596	10023047	16451
Input 2:	#CHR	POS	REF	ALT
	10	64	C	G

```
bedtools annotate -i Input1 -files Input2 -both
```

```
bedtools intersect -wo -a Input1 -b Input2
```

* -wo: write the original A and B entries plus the number of basepairs of overlap between the two features.

closest

find the closest, non-overlapping gene for each interval where
both experiments had a peak
-io ignores overlapping intervals and returns only the closest,
non-overlapping interval (in this case, genes)

```
bedtools closest -a both.bed -b genes.bed -io
```

slop & complement

Step 1. Add 500 bp up and downstream of each probe

```
bedtools slop -i probes.bed -b 500
```

Step 2. Get a BED file of all regions not covered by the probes (+500 bp up/down)

```
bedtools complement -i p.500bp.bed -g hg18.genome
```

window

#Report all genes that are within 10000 bp upstream or downstream of xxx.

```
bedtools window -a xxx.bed -b genes.bed -w 10000
```

Report all genes that are within 10000 bp upstream or 5000 bp downstream of xxx.

```
bedtools window -a xxx.bed -b genes.bed -l 10000 -r 5000
```

#Report all SNPs that are within 5000 bp upstream or 1000 bp downstream of genes. Define upstream and downstream based on strand.

```
bedtools window -a genes.bed -b snps.bed -l 5000 -r 1000
-sw
```

merge

Merge overlapping repetitive elements into a single entry.

```
bedtools merge -i example.bed
```

Merge overlapping repetitive elements into a single entry, returning the number of entries merged.

```
bedtools merge -i example.bed -n
```

Merge nearby (within 1000 bp) repetitive elements into a single entry.

```
bedtools merge -i example.bed -d 1000
```

Random

#Generate random sequences from the genome

```
bedtools random [options] -g <genome>
```

typical options:

-l the length of the interval to generate

-n the number of intervals to generate

The genome file that is supplied takes the form:

<chromName><TAB><chromSize>

For example, Human (hg19):

chr1 249250621

chr2 243199373

...

chr18_gl000207_random 4262

flanking

```
head -n2 genes.bed
```

chr1 134212701 134230065 Nuak2 8 +
chr1 134212701 134230065 Nuak2 7 +

```
bedtools flank -i genes.bed -g ref.chromsizes -l 2000 -r 0 -s > prom.bed
```

This will give you the upstream regions based on strand as follows:

chr1 134210701 134212701 Nuak2 8 +
chr1 134210701 134212701 Nuak2 7 +

```
bedtools getfasta -fi ref.fa -bed prom.bed -fo prom.fa
```

- -l The number of base pairs that a flank should start from orig. start coordinate.
- -r The number of base pairs that a flank should end from orig. end coordinate
- -s Define -l and -r based on strand.
  e.g. if used, -l 500 for a negative-stranded feature,
  it will start the flank 500 bp downstream. Default = false.

How To Use Bedtools To Extract Promoters?