# Comparative Genomics: Finding Orthologs and Paralogs

# Grading / Project Discussion

- Bonus Homework points (e.g. you only have to do 4 homeworks!)

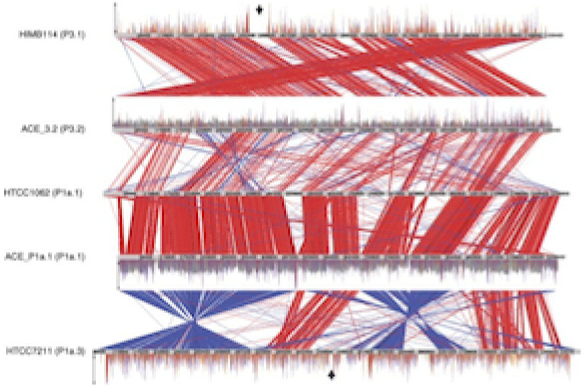
- If you want more Python/BioPython practice problems I am happy to provide…

- Should be getting started on your analysis pipelines

- Commit your script progress to repository.
- Draw/Write out your plans for the steps your tools will perform
- If you are stuck on something, ask sooner.

# Comparative Genomics

- Compare DNA/Genome content
  - Genes
  - Repeats and Transposable Elements
- Compare gene order: Synteny
  - Overall DNA content
  - Gene order



# Repeat Content

Main tool for identifying Repetitive Elements: RepeatMasker

De novo construction of a Repeat Library RepeatModler

See example worked:
https://github.com/biodataprog/code_templates/tree/master/Comparative

```
#!/usr/bin/bash
#SBATCH --ntasks 4 --nodes 1 --mem 16G
module load RepeatModeler
BuildDatabase -name elephant -engine ncbi elephant.fa
RepeatModeler -engine ncbi -pa 4 -database elephant >& run.out
```

This produces a file consensi.fa.classified which can be used as a repeat library

```
>MOLLY_SN#DNA/TcMar-Fot1 RepbaseID: MOLLY_SNXX
acgtacctcacgggttggccggacacacggtttggccggacacttttgcc
aagcccccaccaaattctacctctcaacgtgatgcctcaacaacaacacc
agatagacccttctagcgaacgtcatatacagactgcccttcaagctctt
```

# Repeat Content: RepeatMasker

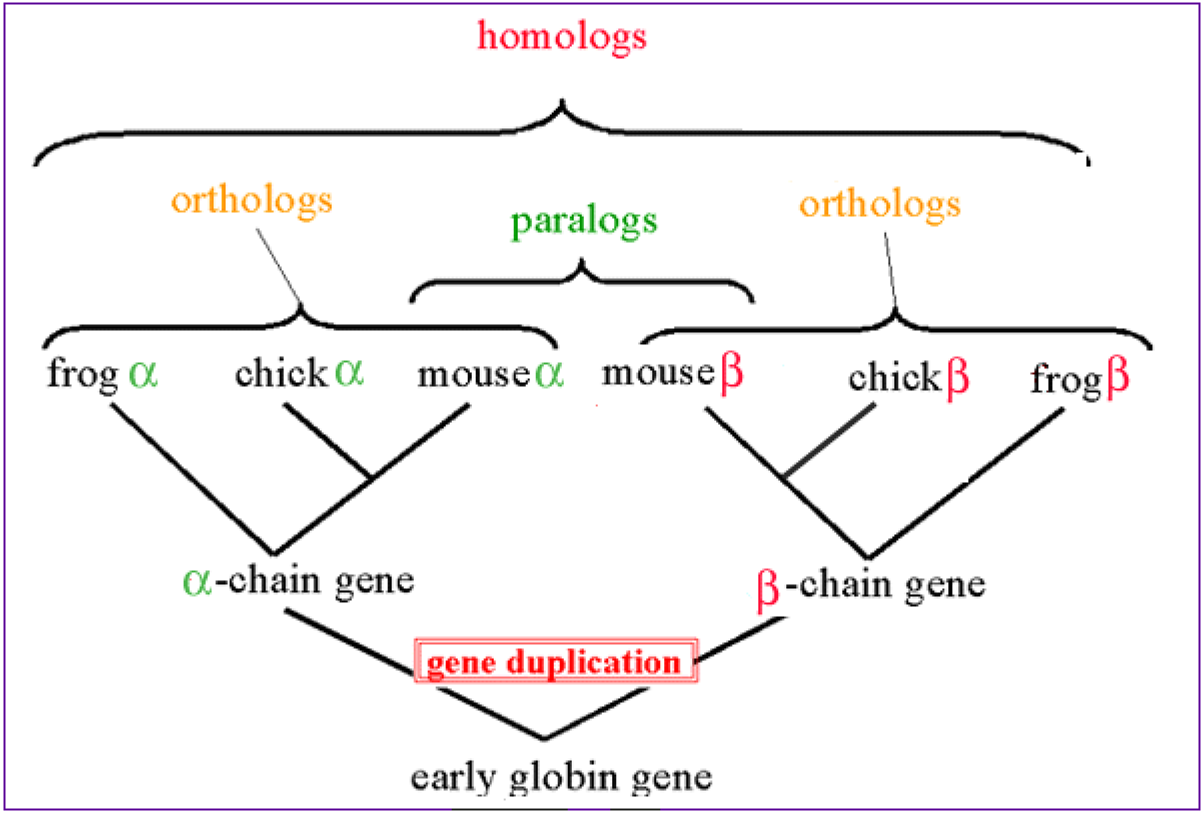
Lots of help here: http://www.repeatmasker.org/webrepeatmaskerhelp.html

```
#!/usr/bin/bash
#SBATCH --ntasks 8 --nodes 1 --mem 16G
module load RepeatMasker
RepeatMasker -lib consensi.fa.classified -pa 8 drosophila.fa
RepeatMasker -species Drosophila -pa 8  -engine ncbi
```

# RepeatMasker Results

```
=================================================
file name: Wolco1.fa
sequences:           348
total length:   50483556 bp  (48243836 bp excl N/X-runs)
GC level:         52.17 %
bases masked:   15644047 bp ( 30.99 %)
==================================================
         number of      length   percentage
         elements*    occupied  of sequence
--------------------------------------------------
LINEs:             1093       685526 bp    1.36 %
      LINE1         127       112478 bp    0.22 %
      LINE2           5         1722 bp    0.00 %

LTR elements:     11045      5410564 bp   10.72 %
      ERV_classI    139        31760 bp    0.06 %
      ERV_classII    60        33459 bp    0.07 %

DNA elements:      4418      1862790 bp    3.69 %
     hAT-Charlie      0            0 bp    0.00 %
     TcMar-Tigger     0            0 bp    0.00 %

Unclassified:     15556      7417912 bp   14.69 %

Total interspersed repeats: 15379533 bp   30.46%
```

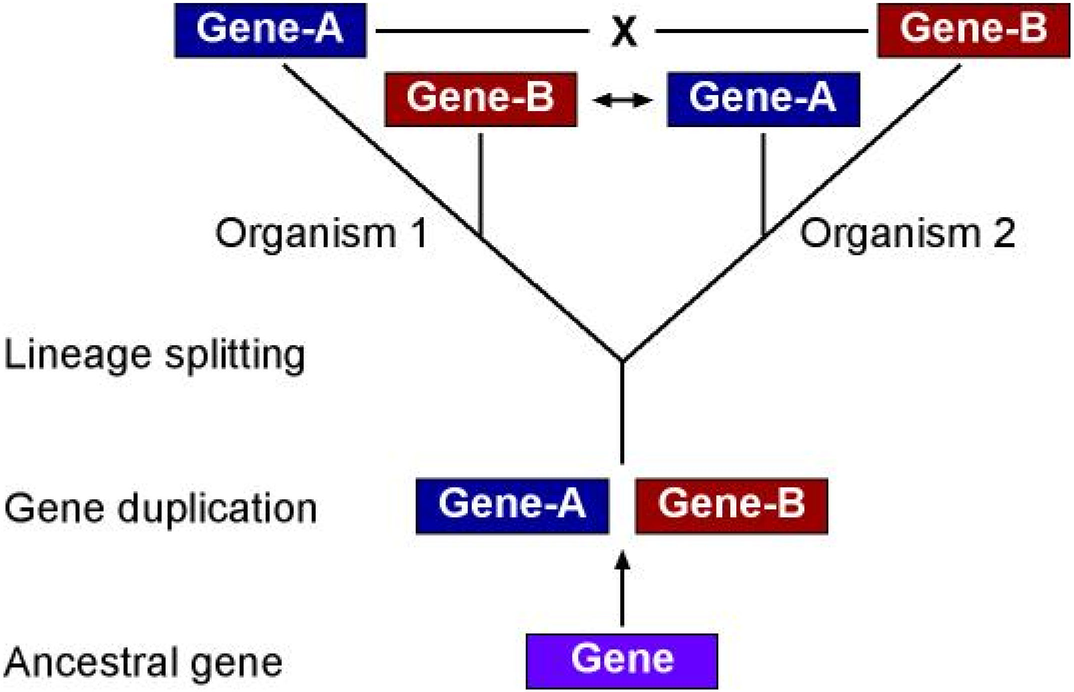
# Orthologs and Paralogs



# Gene families and Orthology

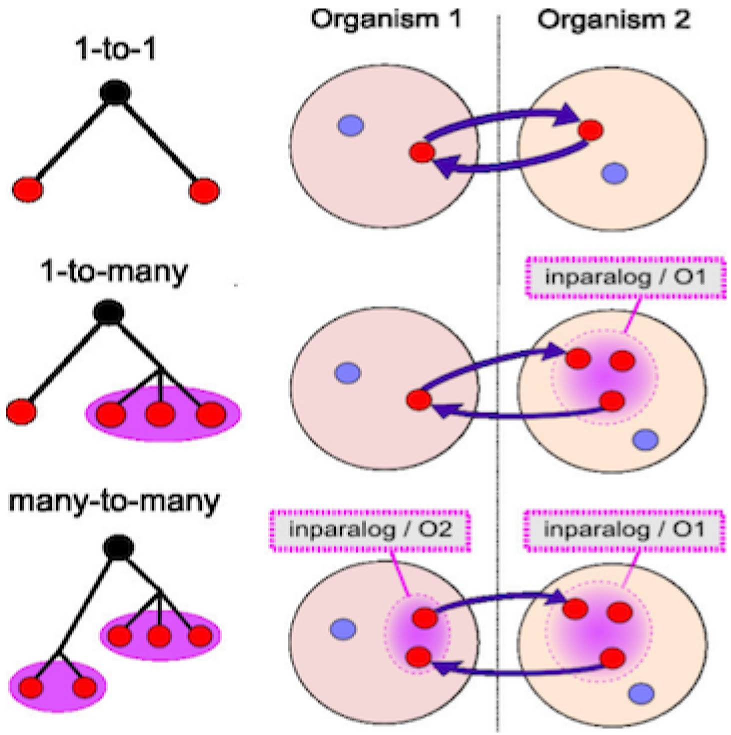
Problem: How to find "same" genes across multiple species.

Genes can duplicate (Paralogs) and can be identical due to descent (Ortholog)



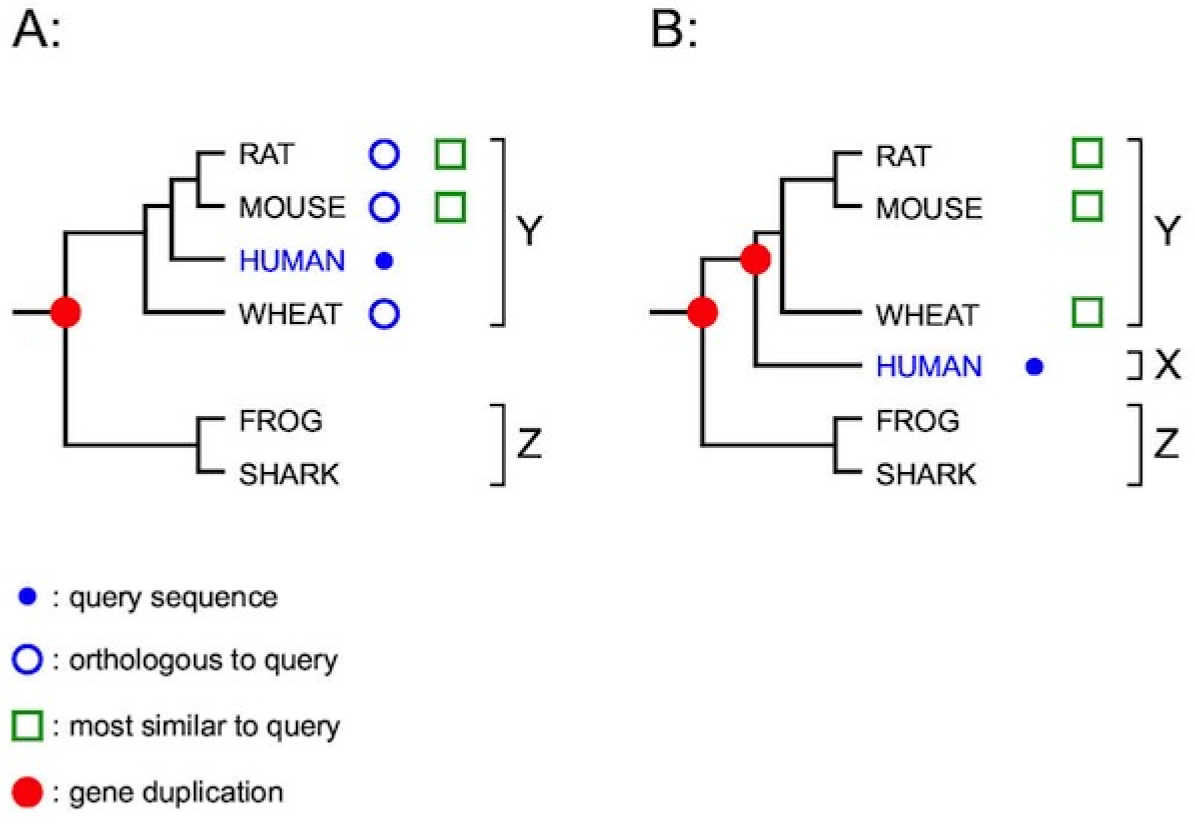
# Methods

- BLAST: 1 way BLAST (Gene A in Species X, what is best hit in Species Y)
- BLAST: reciprocal BLAST



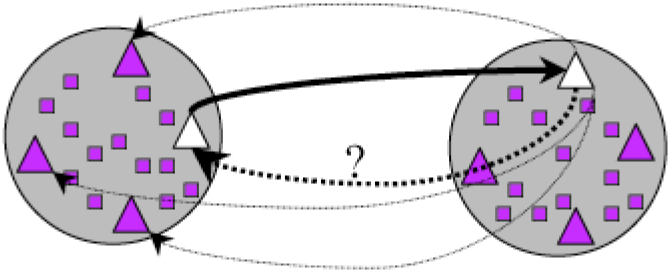
# Trees can help resolve relationships

Best hits can sometimes be wrong (B) though it can be resolved with phylogenetics.



# Reciprocal Searches

- Bi-directional or Reciprocal BLAST



# Implement Bidirectional

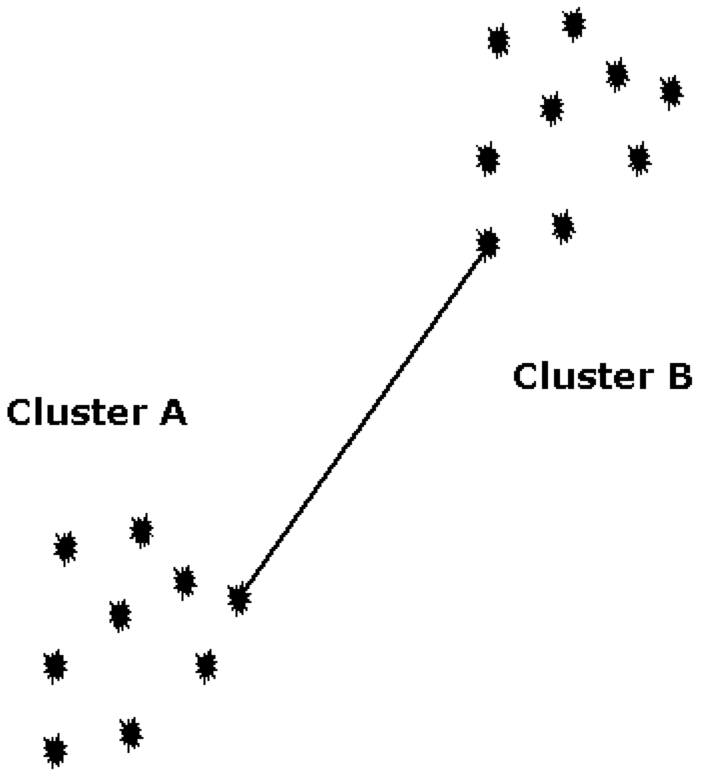
Method to find best top hit in one direction and the reverse.

Let's walk through the code

*Will write this in Python in Class*

# Clustering

- Lumping genes together based on similarity linkage
- Single-linkage means if there is a link between A-B then they are in a cluster

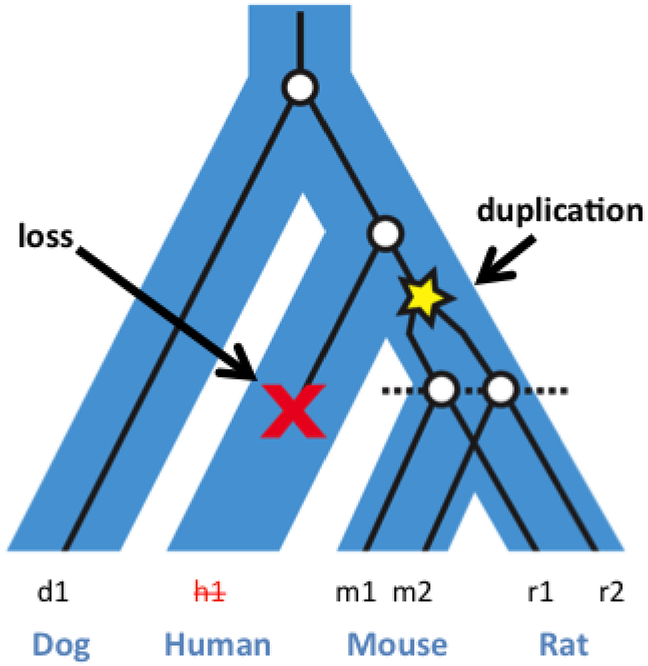


# Code up single-linkage

Let's look at some code.

*Will write this in Python in Class*

# Issues



# Existing solution

- OrthoMCL - requires SQL Database
- Orthagogue - nearly identical results but runs w/o DB

# Steps to build orthologs on cluster

Make sure genome protein FASTA file is

`>SPECIESPREFIX|GENENAME`

```
#!/usr/bin/bash
#SBATCH --ntasks 8 --mem 8G
module load ncbi-blast
CPU=8
cat genome1.pep genome2.pep > proteins.pep
makeblastdb -in proteins.pep -dbtype prot
blastp -query proteins.pep -db proteins.pep -outfmt 6 \
-out proteins_allvsall.BLASTP.tab -num_threads $CPU -evalue 1e-3
module load orthagogue
module load mcl
orthAgogue -i proteins_allvsall.BLASTP.tab -s '|' -e 6 -c $CPU
mcl all.abc -te $CPU --abc -I 1.5 -o orthologs.I15.mcl.out
```

# Ortholog results

```
SP1|GENE1 SP1|GENE2 SP2|GENE3939 ...
Bauco1|Bauco1_125963    Bauco1|Bauco1_427378    Bauco1|Bauco1_562994    CANT|CANT_00
```

# Write script to turn this into a table

```
ORTHOLOG_GRP    SP1   SP2
ORTHO_0001         10     5
ORTHO_0002          1     1
```