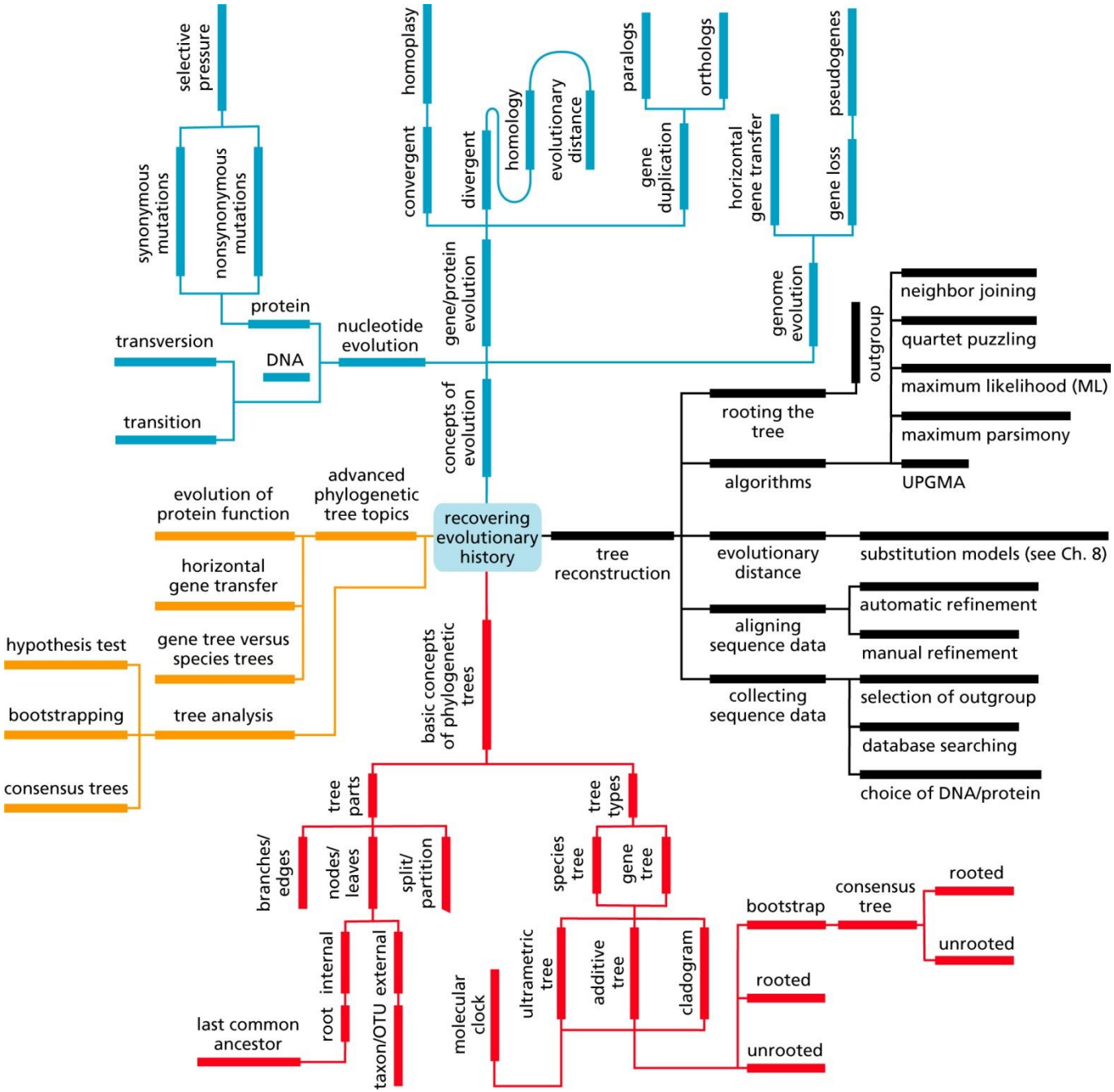
CISC 636 Computational Biology & Bioinformatics (Fall 2016)

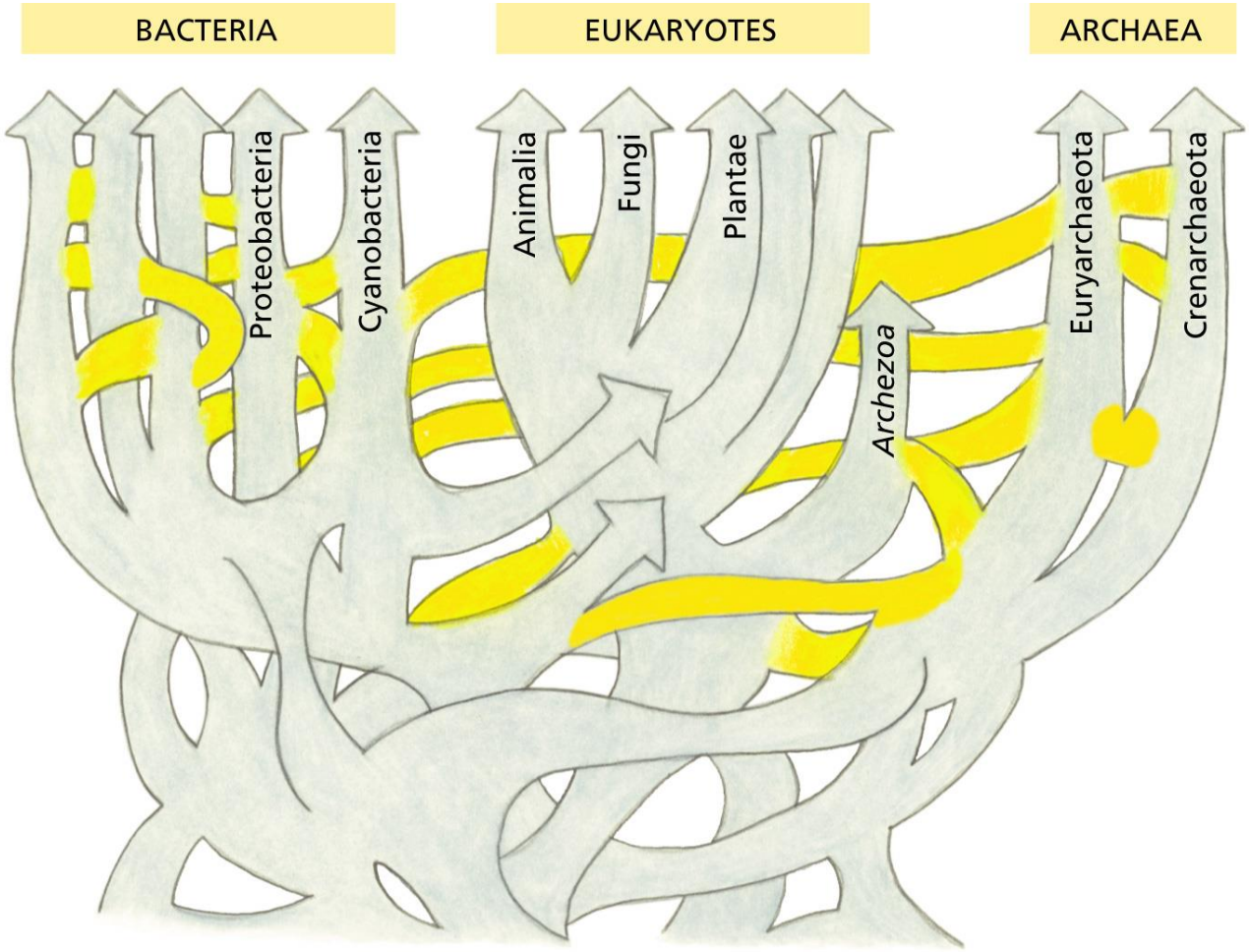
Phylogenetic Trees (I)

Maximum Parsimony

Evolution

- Mutation, selection, Only the Fittest Survive.
- Speciation. At one extreme, a single gene mutation may lead to speciation. [*Nature* **425**(2003)679]
- Phylogeny: evolutionary relation among species, often represented as a tree structure.


selective pressure
synonymous mutations
nonsynonymous mutations
protein
DNA
transversion
transition
nucleotide evolution
homoplasy
convergent
divergent
homology
evolutionary distance
paralogs
orthologs
gene duplication
horizontal gene transfer
pseudogenes
gene loss
gene/protein evolution
genome evolution
concepts of evolution
neighbor joining
quartet puzzling
maximum likelihood (ML)
maximum parsimony
UPGMA
outgroup
rooting the tree
algorithms
evolution of protein function
advanced phylogenetic tree topics
recovering evolutionary history
tree reconstruction
evolutionary distance
substitution models (see Ch. 8)
horizontal gene transfer
gene tree versus species trees
hypothesis test
bootstrapping
consensus trees
tree analysis
aligning sequence data
automatic refinement
manual refinement
collecting sequence data
selection of outgroup
database searching
choice of DNA/protein
basic concepts of phylogenetic trees
tree parts
branches/ edges
nodes/ leaves
split/ partition
root
internal
external
taxon/OTU
last common ancestor
tree types
species tree
gene tree
molecular clock
ultrametric tree
additive tree
cladogram
bootstrap
consensus tree
rooted
unrooted
rooted
unrooted

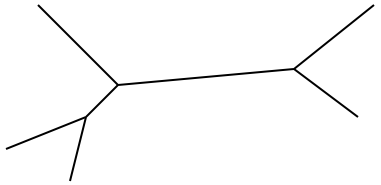

BACTERIA
EUKARYOTES
ARCHAEA
Proteobacteria
Cyanobacteria
Animalia
Fungi
Plantae
Archezoa
Euryarchaeota
Crenarchaeota

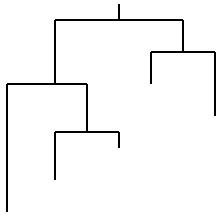
Question: how to infer phylogeny?

- Based on morphological features
- Based on molecular features
  - Gene trees
  - Phylogenetic trees (using 16s rRNA)
  - Criteria for selecting features
    - Ubiquitous
    - Relatively stable
  - Reconciliation between gene trees and species trees
    - Orthology genes

Trees (binary)

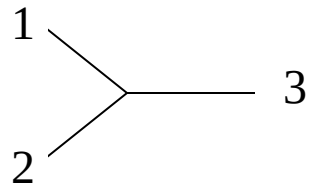
- Unrooted vs rooted

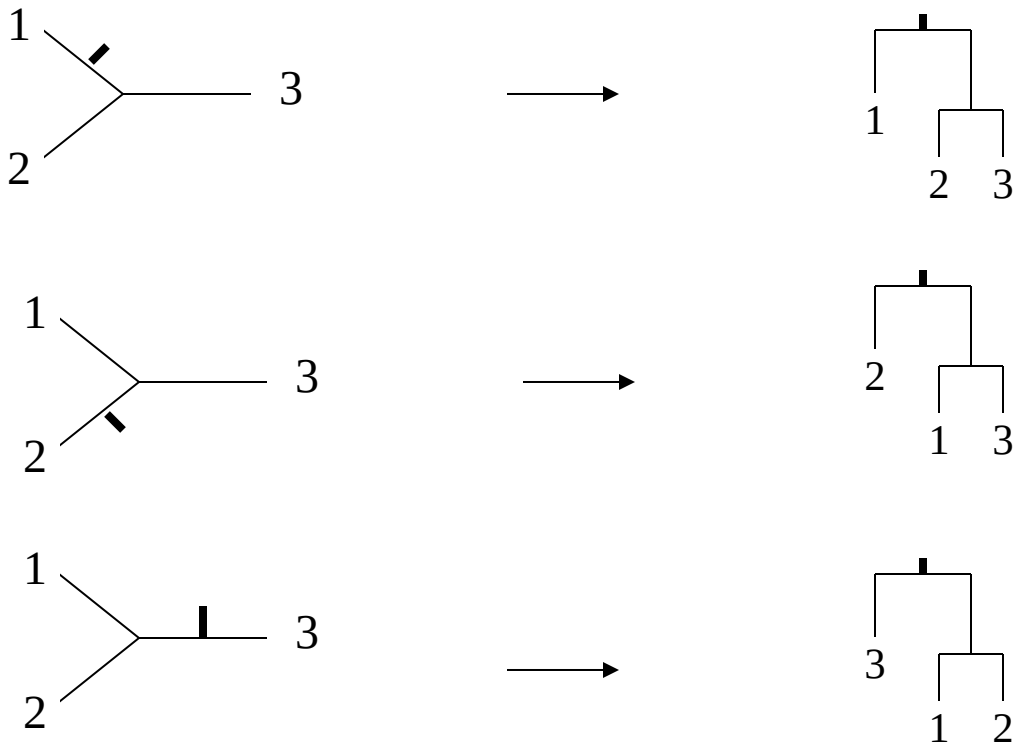




- Leaves versus internal nodes
- For an unrooted binary tree with n leaves
  - # of nodes (including leaves) is 2n – 2.
  - # of edges is 2n -3
  - Can lead to 2n-3 rooted trees, by adding a root at any edge.

- For example,





How many different configurations can a tree of n leaves have?

Assume the tree is unrooted.

Grow the tree by adding one leaf at a time

n = 2, there is 1 edge to break.

n = 3, there are 3 edges to break => 3 different configurations

n = 4, there are 5 edges to break => 5 different configurations

…

n = n, there are (2n-3) edges to break => (2n-5)

$1 \cdot 3 \cdot 5 \cdot 7 \cdot \ldots \cdot (2n-3) = (2n-3)!!$

The number of possible configurations as a function of the tree size increases very fast.

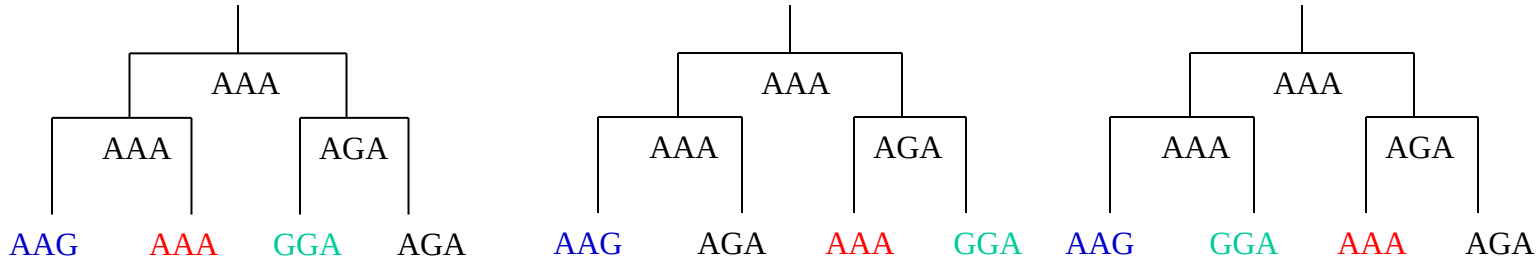
Parsimony

- Based on sequence alignment.
- Assign a cost to a given tree
- Search through the topological (configuration) space of all trees for the best tree: the one that has the lowest cost.

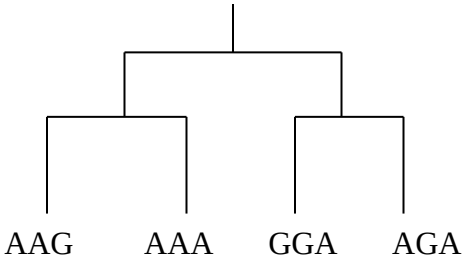
For example, given an alignment of four sequences

AAG
AAA
GGA
AGA

If the number of mutations is used as a measure of cost, then the leftmost tree in the following is the best tree.



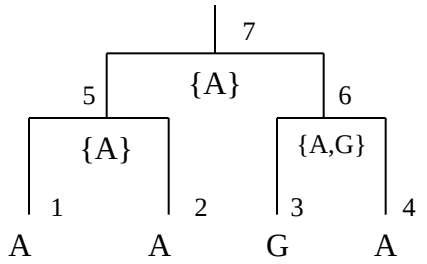
Algorithm: unweighted parsimony [Fitch 1971]
// given an alignment A of n sequences
// each position in A is treated independently
// Tree T with n leaves labeled for each sequence



C = 0; // the total cost

for (u = 1 to |A|) { // u is the position index into the alignment A
initialization:
set C_u = 0 and k = 2n -1 // C_u is the cost and k is the node index
// index starting 1, from left to right, bottom to up

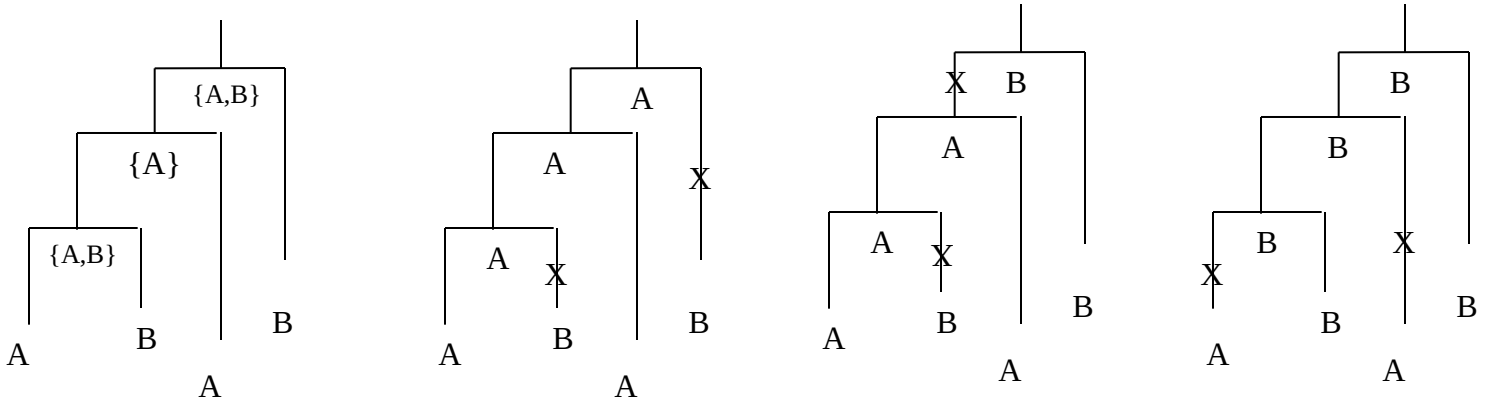
recursion: to obtain the set R_k // contains candidate residues assigning to node k
if k is leaf node:
set $R_k = x_u$ // which is the residue at position u
else
compute R_i, R_j for the daughter nodes i, j of k
if ($R_i \cap R_j$) is not empty:
set $R_k = R_i \cap R_j$
else
set $R_k = R_i \cup R_j$
$C_u = C_u + 1$
termination: $C = C + C_u$
}
minimal cost of tree = C.



Trackback phase:

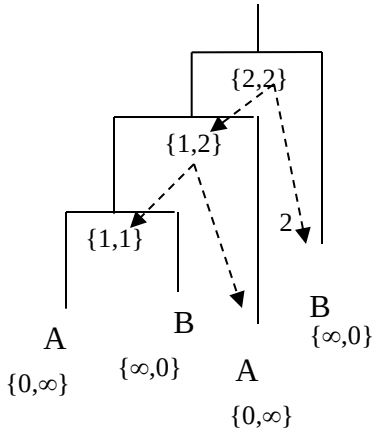
- Randomly choose a residue from R_{2n-1} (the root) and proceed down the tree.
- if a residue is chosen from the set R_k
  - Choose the same residue from the daughter set R_i if possible, otherwise pick a residue at random from R_i.
  - Choose the same residue from the daughter set R_j if possible, otherwise pick a residue at random from R_j.

For example,



Traceback cannot find this tree, although it is equally optimal as the other two trees.

Algorithm: Weighted parsimony [Sankoff & Cedergren 1983]



// given an alignment A, each position in A is treated independently
// Tree T with the leaves labeled, and a residue substitute score matrix S.

C = 0; // the total cost

for (u = 1 to |A|) { // u is the position index into the alignment A

initialization:

set k = 2n -1 // k is the node index, currently pointing to the root

recursion: Compute $S_k(a)$ // the minimal cost for assigning residue a to node k

if k is leaf node:

if a = x_u^k then $S_k(a) = 0$

else $S_k(a) = \infty$ // cannot substitute a leaf

else // k is not a leaf node

compute $S_i(a)$, $S_j(a)$ for all a at the daughter nodes i, j of k

set $S_k(a) = \min_b [S_i(b) + S(a,b)] + \min_b [S_j(b) + S(a,b)]$

set $l_k(a) = \text{argmin}_b [S_i(b) + S(a,b)]$, $r_k(a) = \text{argmin}_b [S_j(b) + S(a,b)]$. // for traceback

termination:

$C = C + \min_a S_{2n-1}(a)$.

}

minimal cost of tree = C.

- Both algorithms run in O(nm), where n is number of sequences and m is the sequence length in terms of number of residues.

- Weighted parsimony, when using S(a,a) = 0 for all a and S(a,b) = 1 for all a ≠ b, gives the same cost as that for the traditional parsimony.

- Traceback in weighted parsimony can find assignments that are missed in the traditional unweighted parsimony.

- The cost from the unweighted parsimony is independent of the position for the root node. Therefore, the cost can be computed using unrooted trees.

- Still the number trees to search using parsimony grows huge as the number of leaves increases. It is proved that finding the most parsimonious tree is an NP-hard problem.

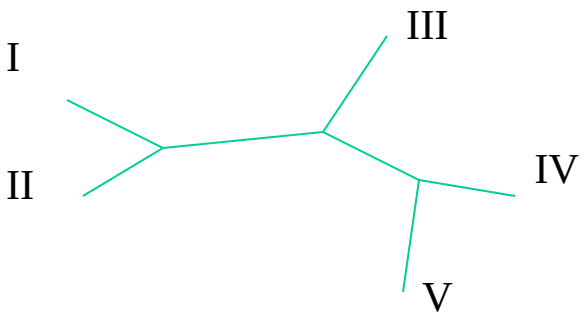
- Branch-and-bound
  - Guarantee to find the optimal tree
  - Worse-case complexity is the same as exhaustive search.

Assessing the trees: the ***bootstrap***

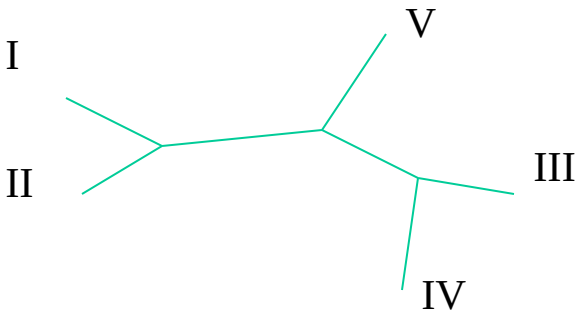
- "Plug-in" sampling with replacement
  - Given an alignment with, say, one hundred columns.
  - Randomly select one column from the original alignment as the first column, and repeat this process until one hundred columns are selected forming a new alignment of one hundred columns.
  - Use this artificially created alignment for parsimony analysis, a new tree is found.
  - Repeat this whole process many times (say 1000).
  - The frequency with which a chosen phylogenetic feature appears is used as a measure of the confidence we have in this feature.

Bootstrap

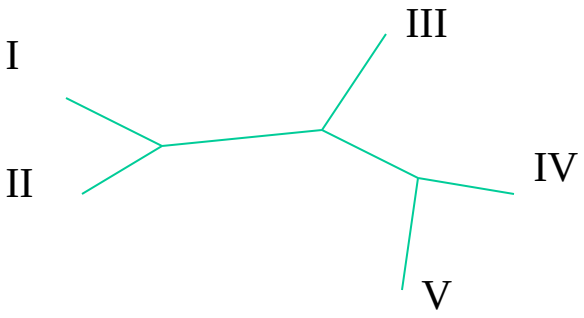
```
     1234567890
I    GGGGGGATCA
II   GGGAGTATCA
III  GGATAGACAT
Iv   GATCATGTAT
V    GTTCATATCT
```

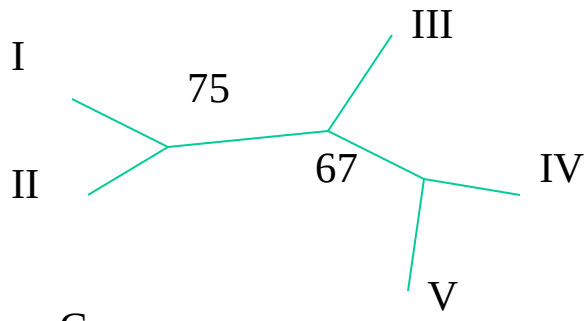



```
     1155569999
I    GGGGGGCCCC
II   GGGGGTCCCC
III  GGAAAGAAAA
Iv   GGAAATAAAA
V    GGAAATCCCC
```



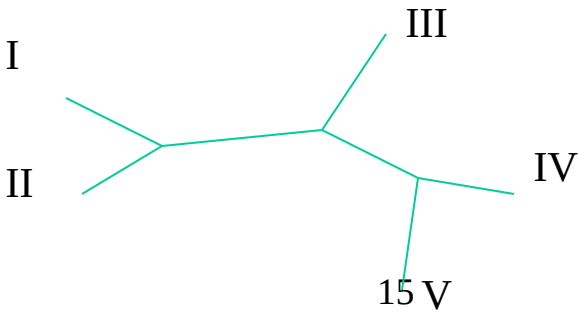

```
     1224555770
I    GGGGGGGAAA
II   GGGAGGGAAA
III  GGATAAAAAT
Iv   GATCAAAGGT
V    GTTCAAAAAT
```





Consensus bootstrap tree

```
     3334667888
I    GGGGGGATTT
II   GGGATTATTT
III  AAATGGACCC
Iv   TTTCTTGTTT
V    TTTCTTATTT
```



Software packages and databases for phylogenetic trees

- Phylip by Felsenstein (http://evolution.genetics.washington.edu/phylip.html)
- PAUP (http://paup.csit.fsu.edu/)
- MacClad (http://macclade.org/macclade.html)
- TreeBase (http://www.treebase.org/treebase/)