Phylogeny

Terminology

A **phylogeny** is the evolutionary history of an organism

A **taxon** (*plural:* **taxa**) is a group of (one or more) organisms, which a taxonomist adjudges to be a unit.

A definition? from Wikipedia

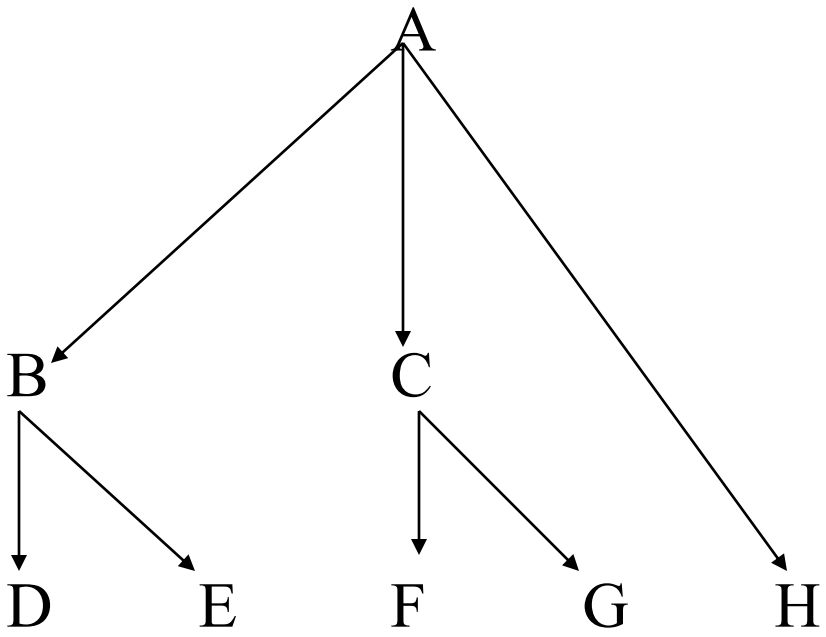
A natural next step from MSA…

A	T	G	C	T	G	A	FLATWORM
A	$-$	G	G	T	G	A	ROUNDWORM
$-$	T	T	C	T	G	A	SNAIL
A	A	T	G	A	G	T	CLAM
T	A	T	G	A	C	T	BARNACLE

Phylogeny

A phylogenetic tree is a way to portray changes during evolution

- Leaves are the individuals (D,E,F,G,H)
- Interior nodes are ancestors (A,B,C)
- Edges may be labeled to represent evolutionary distances



The Computational Problem

We are given current or recent observations of species.

In addition, perhaps we may be able to know evolutionary distances (time) or perhaps numbers of mutations

Our task is to infer the phylogenetic tree

Tree Building Problem

There are $\frac{(2n-3)!}{2^{n-2}(n-2)!}$ different ways to build a rooted tree.

This is a hard problem. The table below gives a quick insight into the enormity

Number of nodes/leaves	**Number of possible trees**
4	15
9	~2,000,000
20	8×10^{21}
50	$\sim 3 \times 10^{76}$

Complexity

Phylogeny construction is a hard problem. To solve it, one must invoke the usual techniques

- Branch-and-bound
- Heuristics such as simulated annealing or genetic algorithms
- Sometimes there are special cases which make it tractable

Kinds of Problems

- Character
  - Character is an n-ary state
  - Seldom or slow changing characters such as morphology (number of legs, shape of wing)
- Distance
  - Measure is evolutionary time- edges are labeled
- Maximum Likelihood
  - Computes most probable tree

Character Based Data

- Characters are n-ary
- Characters change seldom or slowly, such as morphology (number of legs, shape of wing)
- Strategy: Partition the ‘leaves’ in order to minimize the edit distance

Character Based Issues

Problems that can occur:

- Convergent Evolution
  - Two or more taxa have the same value (state) for the same character. The assumption would be that the taxa are close, but convergent evolution could also account for this
- Reversals
  - If a taxon inherits the state of a character from an immediate ancestor having that character in a positive state, but a related taxon does not inherit that character from the very same ancestor, the character state is ‘lost’ in that second taxon
  - Suppose there is another (not common) ancestor, in which the character state is negative. An immediately descendent taxon, in which the character state is positive, has ’gained’ the state.
  - The trouble has arisen from the reversed states of the character when comparing the two immediate ancestors

The remedy is either to accept these problems or to assume (or actually verify) a perfect phylogeny

Character Based Phylogeny: Perfect Phylogeny

A Perfect Phylogeny:

For each character, when a transition of state for a given character occurs along an edge of the tree, every node in the sub-tree below that edge has the same (new) state for that character

Under certain circumstances, finding and constructing the tree of a perfect phylogeny is tractable, despite the intractability of finding trees in general.

So… Two Issues

1. Is the phylogeny perfect?

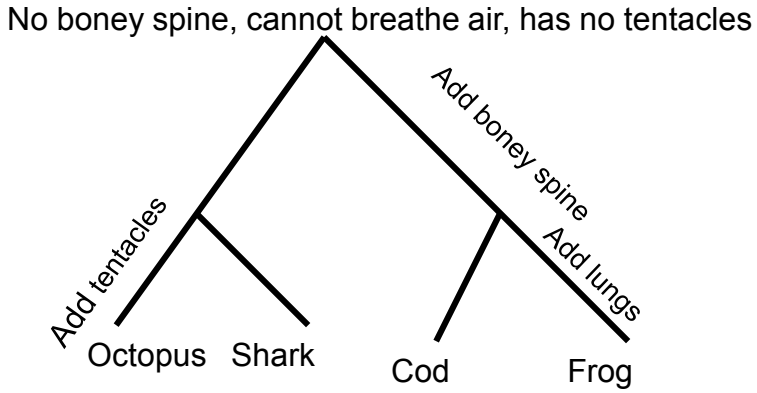
2. If so, is the tree construction a tractable problem?

A Perfect Phylogeny: Example

Consider these creatures:

	Boney Spine	Breathes Air	Tentacles
Octopus	N	N	Y
Shark	N	N	N
Cod	Y	N	N
Frog	Y	Y	N

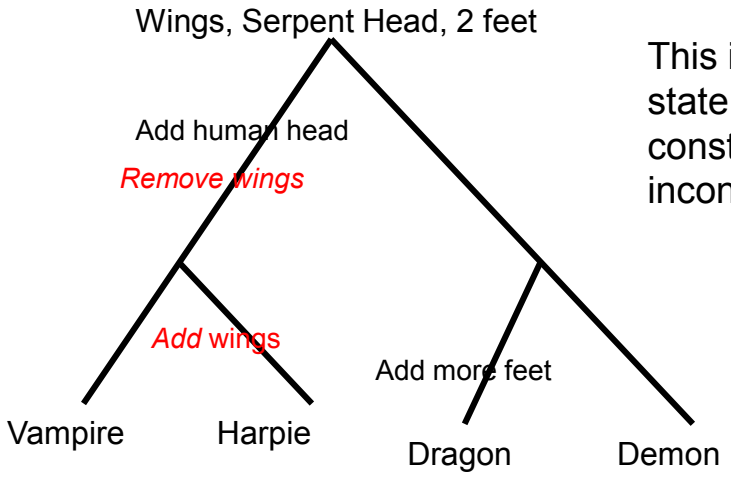
Here is a perfect phylogeny:



For each character, once a state is achieved, it is preserved throughout the sub-tree

Imperfect Phylogeny: example

	Head	Feet	Wings
Demon	Serpent	2	Y
Harpie	Human	2	Y
Dragon	Serpent	4	Y
Vampire	Human	2	N



This is NOT a perfect phylogeny; the state of the wings character is constant in the right sub-tree but inconstant in the left sub-tree

Perfect Phylogeny Solution

The determination of whether inferring a perfect phylogeny is tractable depends on :

- Whether the characters are *ordered* and *directed*, that is, whether the change from state a to state b always occurs in the same order, and in the same direction. If so, the problem is tractable.

- If there are only two possible states (binary), then the only remaining issue, whether the transition is *directed,* becomes moot. The problem is tractable.

Otherwise, the solution to even a perfect phyogeny is NP-hard

Perfect Phylogeny Solution

A special case: The tree is binary

- Is it a perfect phylogeny?
  - There is a quadratic algorithm to answer yes or no
- If so, what is the solution?
  - There is a quadratic algorithm to build the tree

Binary Tree: Is it a Perfect Phylogeny? A Polynomial Algorithm

- Create a Binary Matrix: Characters are columns, taxa are rows

- Designate the state of each character by 1 or 0.
  - The convention (WLOG) is that the root is all 0's
  - Create a set for each character (*sci* column) whose elements are the IDs of the taxa that possess that character

- By Theorem:

  For all sets, taken pair-wise , if each set is either a proper subset of the other, or if their intersection is empty, then the phylogeny is perfect.

A Polynomial Algorithm: Example

	Boney Spine	Breathes Air	Tentacles
O Octopus	N	N	Y
S Shark	N	N	N
C Cod	Y	N	N
F Frog	Y	Y	N

Express this matrix as a binary matrix

	c_1	c_2	c_3
O	0	0	1
S	0	0	0
C	1	0	0
F	1	1	0

Here are the taxa with a 1 included in the set for that column:

c_1	c_2	c_3
C,F	F	O

Here are the relations of the sets taken pair-wise:

$c_2 \subseteq c_1$

$c_1 \cap c_3 = \varnothing$

$c_2 \cap c_3 = \varnothing$

Conclusion: Theorem applies→phylogeny is perfect

Building the Tree

- The tree for a perfect phylogeny can be built in polynomial time

- The recipe:
  - Start with the root
  - Find a character in state 1
  - Create an node
  - Create an edge back to the current node with the character as the label
  - Make the new node current
  - If the edge exists, continue

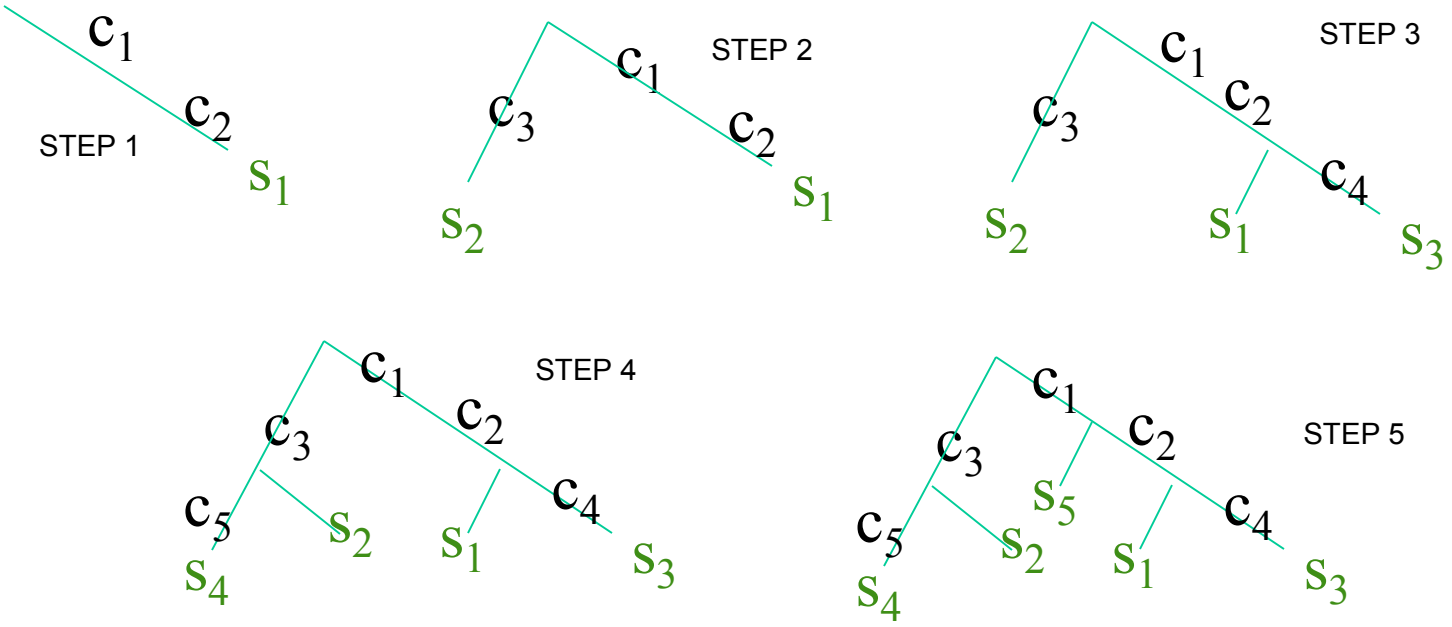
Building* the Tree

Start with the binary matrix sorted by column, with row 1 as the MSB

	c_1	c_2	c_3	c_4	c_5
s_1	1	1	0	0	0
s_2	0	0	1	0	0
s_3	1	1	0	1	0
s_4	0	0	1	0	1
s_5	1	0	0	0	0

*Example from Gusfield

Follow the recipe as previously described (follow the one's in each matrix row)



Character Based Phylogenies

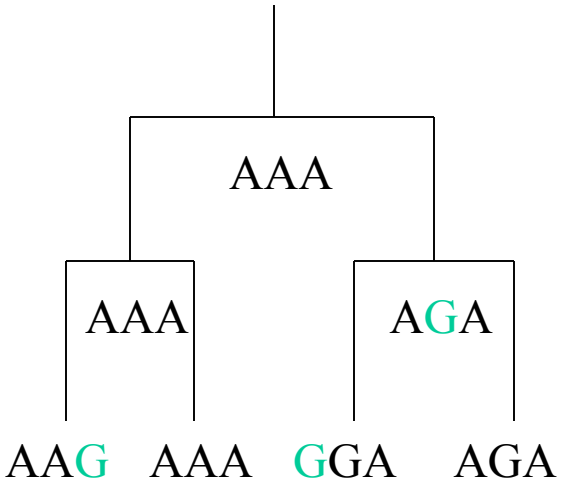
- We have established that there are an exponentially growing number of possible topologies for increasing number of leaves.
  - This is the *Phylogeny Problem*

- *Given a specific topology*, we need a way to guarantee that the internal nodes are determined in such a way that the minimal edit distance is embodied in the arrangement of nodes and leaves
  - This is called *The Small Phylogeny Problem*

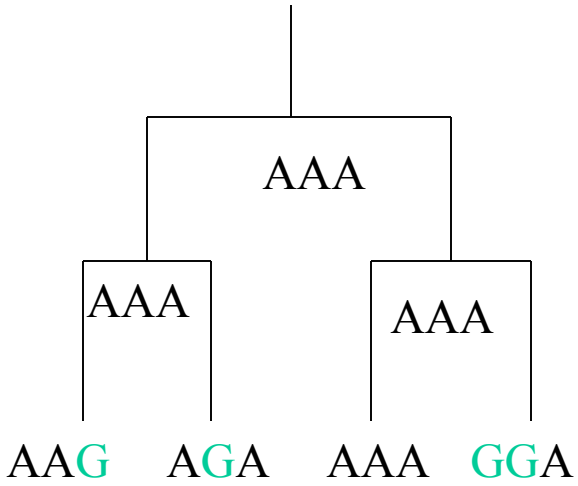
The (Large) Phylogeny Problem

Consider 3 characters

AAG,AAA,GGA,AGA



Score 3



Score 4

The score (total Hamming distance of all characters) of the tree depends on the order of the leaves and the topology of the tree

Parsimony

While we may need a heuristic for the Phylogeny Problem, **the Small Phylogeny Problem can be solved in quadratic time by Dynamic Programming**

Fitch's Parsimony Algorithm minimizes the total of the edit distances as it builds the tree

Strategy:

- ✓Bottom Up
  - Label the internal nodes
- ✓Top Down
  - Score the tree

Fitch's Parsimony Algorithm

- The bottom up phase sets the possible alternatives for internal nodes and allows for scoring

- The top down phase optimally selects the internal nodes from the available choices and can also be used for scoring

- Score is determined for each separate character. One may score from either bottom up or top down, but use one or the other, not both

    - It is useful to score bottom up, then use the top down score as a validation of accuracy, particularly if building by hand, where an error is much more likely

Heuristic
Fitch's Parsimony Procedure

- Bottom -Up phase
  - If the children of an as yet unlabeled node have a common possible state, the parent takes that common state
  - Otherwise give the node all possible states of its children – pay 1 point penalty
- Arbitrarily pick a state for the root if there is more than one choice

- Top-Down phase
  - If the set of all possible child nodes from a parent contains the assignment of the parent (*ie* same character) , assign the state of the parent to the node
  - Otherwise choose one of the possible states arbitrarily and pay 1 point penalty

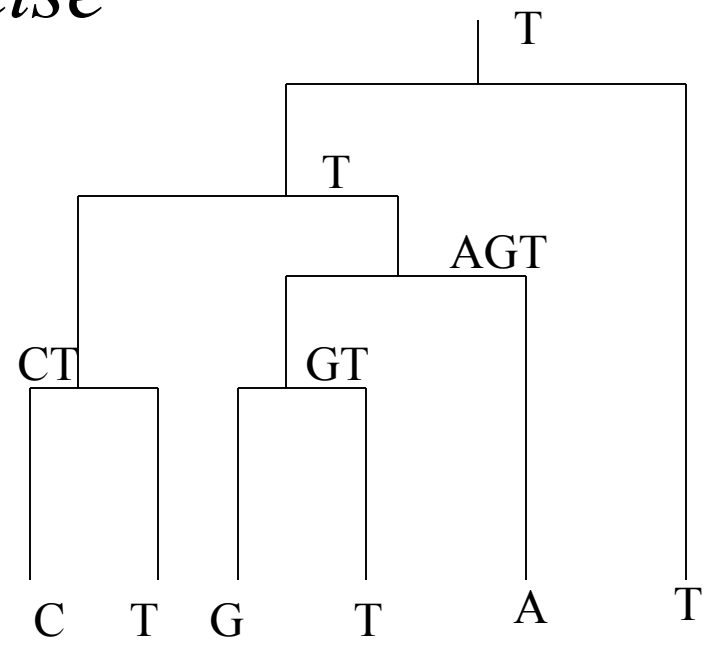
Fitch's Algorithm
Bottom Up (Leaves→Root)

One character is shown in this example

States R_i of node i with children j,k

$$R_i = \begin{cases} R_j \cup R_k \ if \ R_j \cap R_k = \varnothing \\ R_j \cap R_k \ else \end{cases}$$

1 point penalty for every union

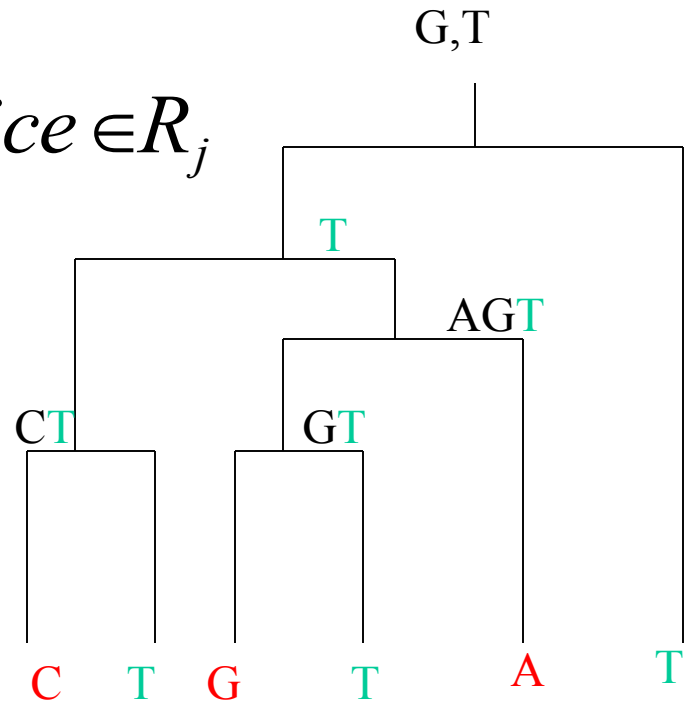


Fitch's Algorithm
Top Down (Root→Leaves)

For node j, with parent i, select one of the possible states ($r_i \in R_j$). Then

$$r_i = \begin{cases} r_i, \text{ if } r_i \in R_j \\ \text{else arbitrary choice} \in R_j \end{cases}$$

1 point penalty for every arbitrary choice



In Plain English..

Going up, for each character….

- If there is a common state, keep only it
- Otherwise, keep all the states and pay* a point

Going down, for each character….

- Keep the common state between parent and child, if there is one
- If not, pick one from the child arbitrarily and pay* a point

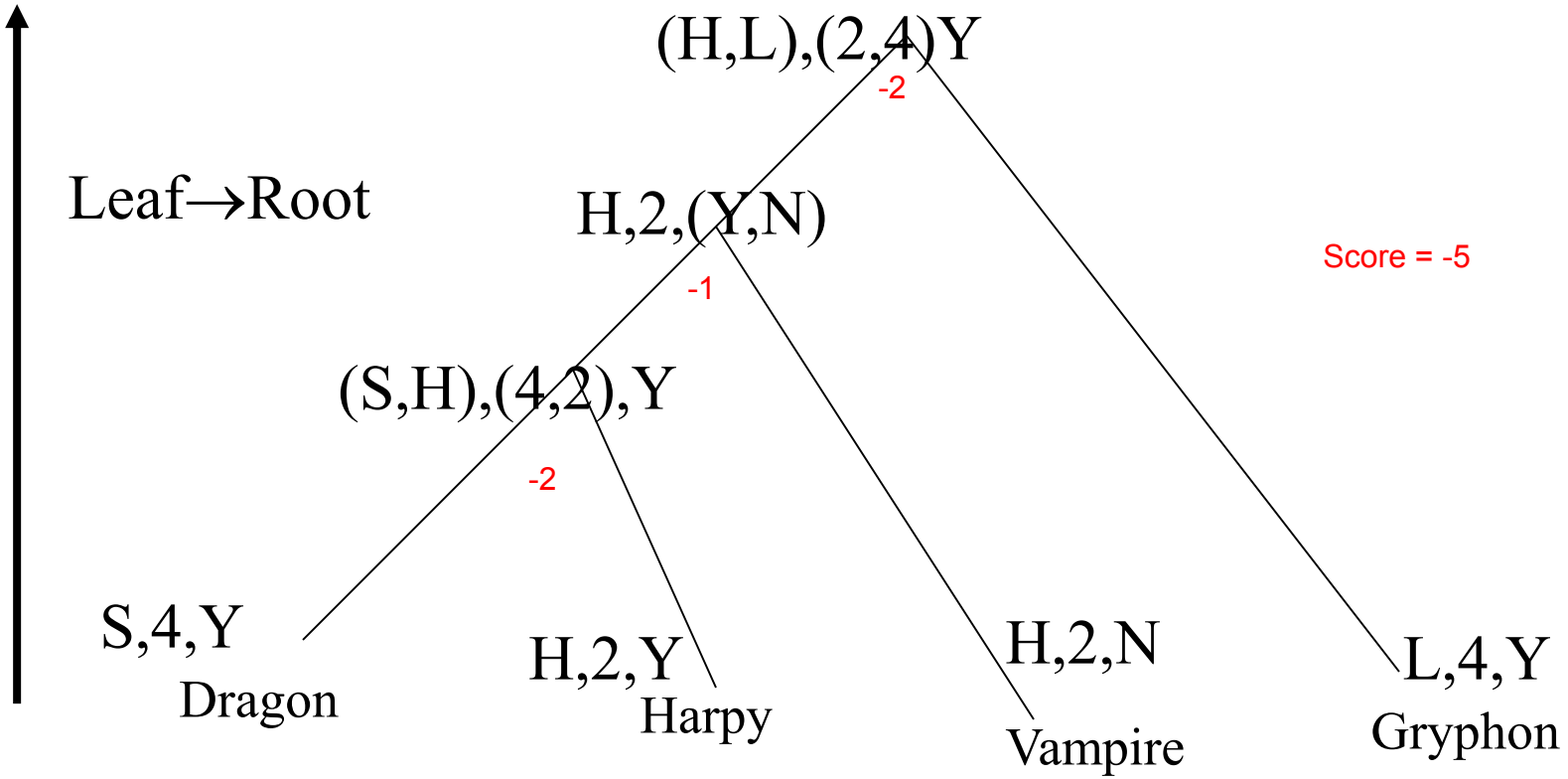
*The penalty is paid either going up or going down, but not twice

Legendary Creatures With Fangs

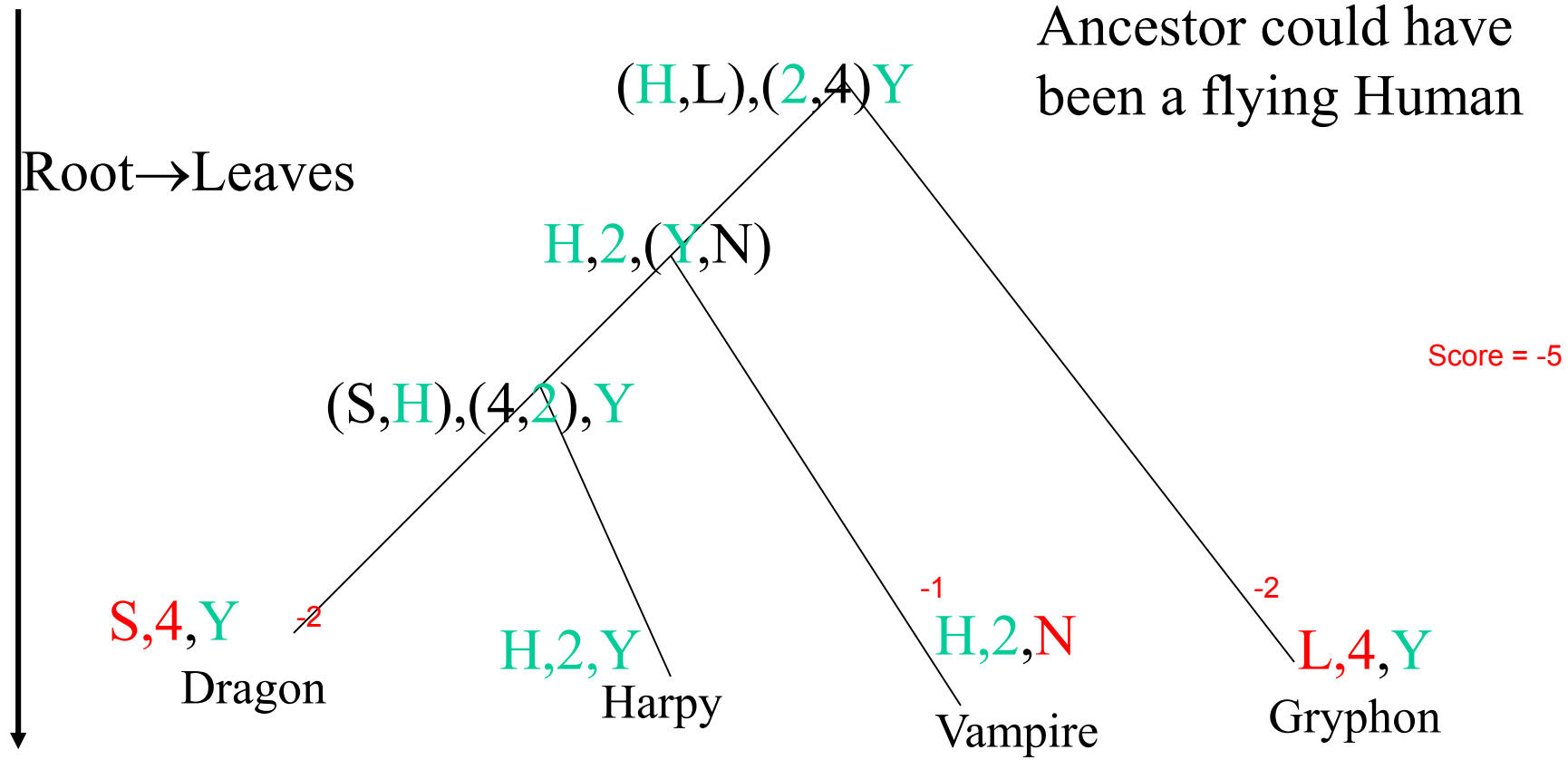
	Head	Feet	Wings
Gryphon	Lion	4	Y
Harpie	Human	2	Y
Dragon	Serpent	4	Y
Vampire*	Human	2	N

*In non-bat form

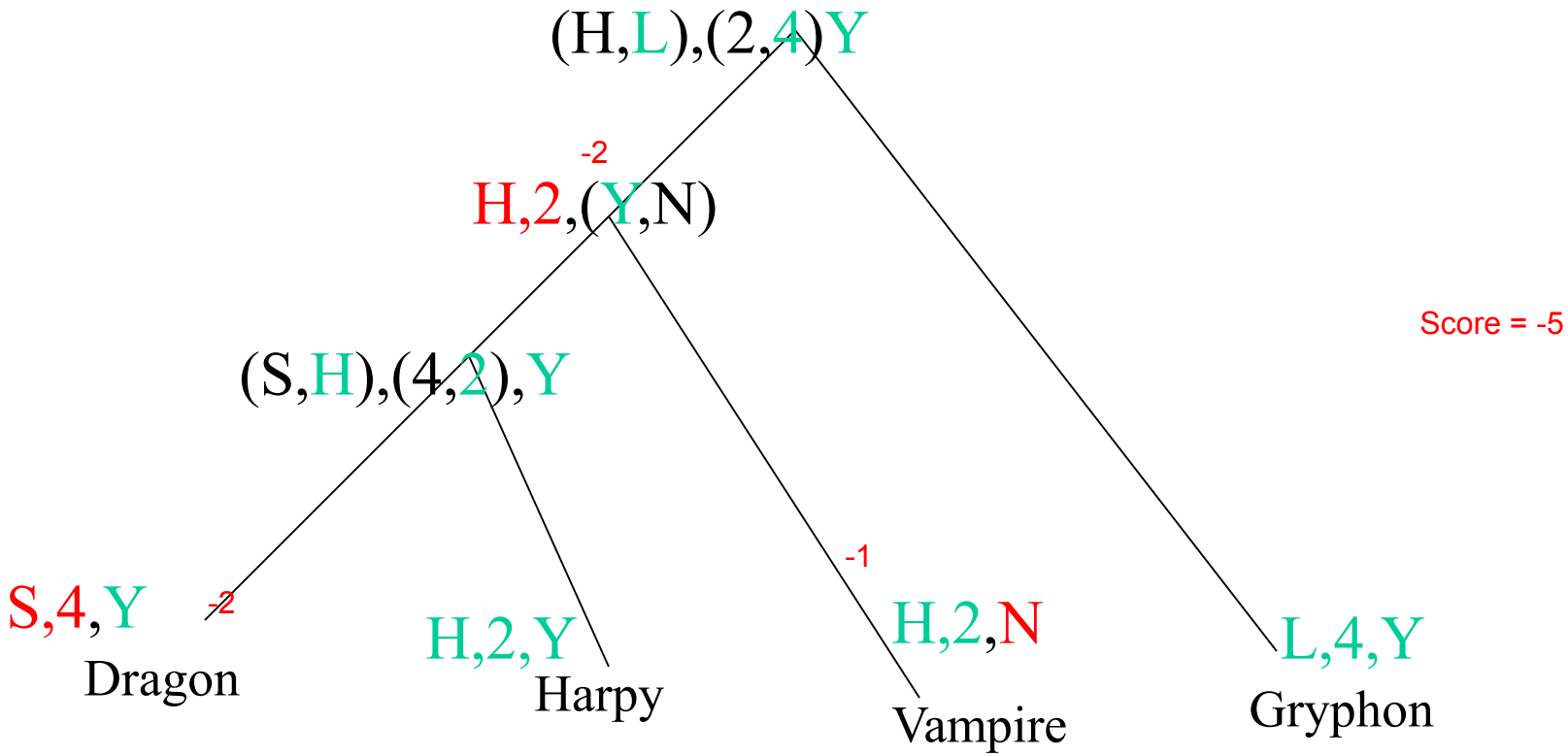
Heuristic
Fitch's Algorithm



Fitch's Algorithm



On the other hand, it could have been a Gryphon, a 4-legged flying Human, or a 2-legged flying Lion



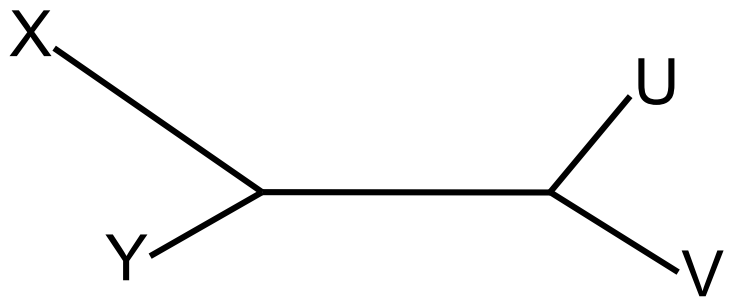
Distance Based Phylogeny Computational Task

- Leaves are events/observations
- Edge labels are evolutionary distances
- We are given distances between nodes in a distance matrix, or we can create one

Task: We must infer not only the tree topology but the appropriate edge labels

About Distance Trees

Additivity is an attribute that enables finding a tree in polynomial time



If the following holds for any 4 points in the tree, the tree is additive:

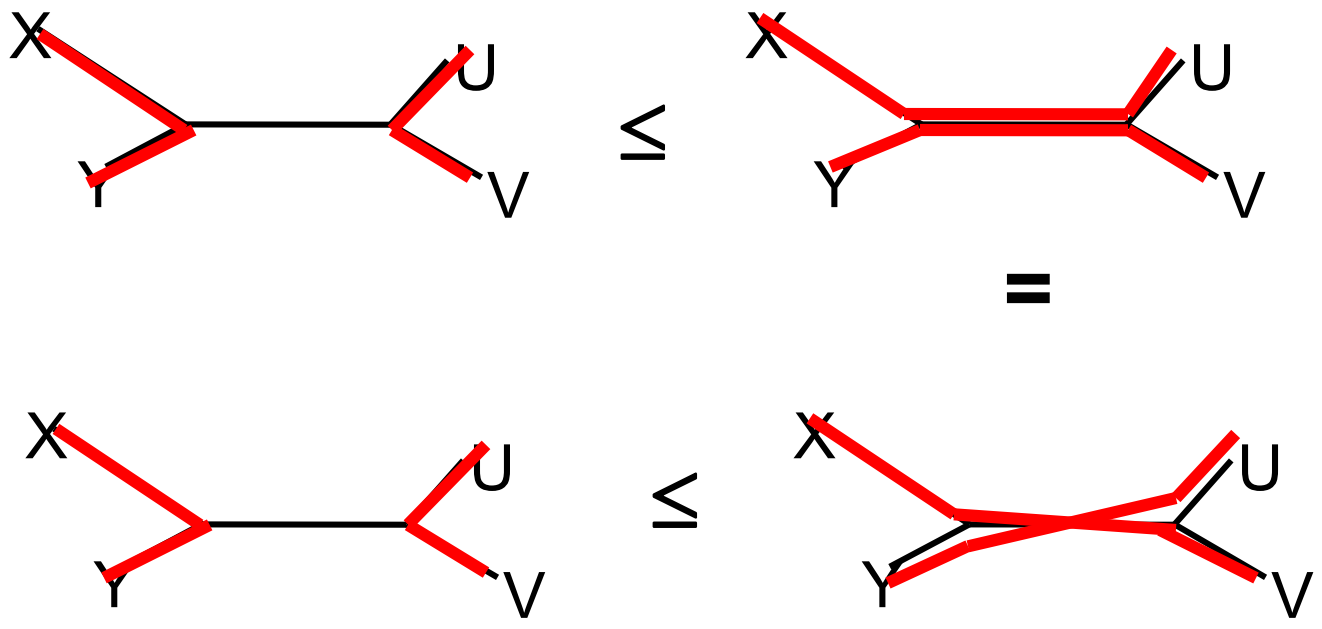
$$d(x,y)+d(u,v) \le d(x,u)+d(y,v)$$

$$d(x,y)+d(u,v) \le d(x,v)+d(y,u)$$

$$d(x,u)+d(y,v) = d(x,v)+d(y,u)$$

4-point Condition

In summary



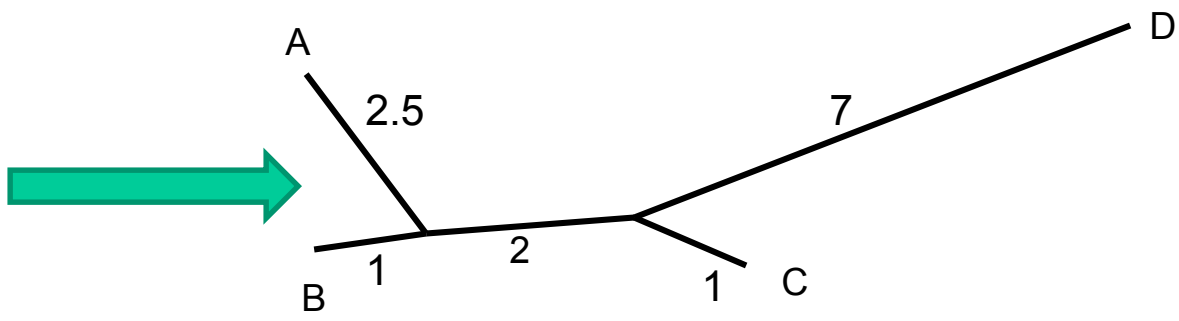
Distance Based

- Form a distance matrix such that each row and column corresponds to a species and the entries are the distances between
- Find a tree such that the distance between each pair of leaves is equal to the distance between these species as given in the distance matrix
- The distance matrix must be additive to give a **polynomial solution**

ADDITIVE MATRIX

	A	*B*	*C*	*D*
A	0	3.5	5.5	11.5
B	3.5	0	4	10
C	5.5	4	0	8
D	11.5	10	8	0

ADDITIVE TREE

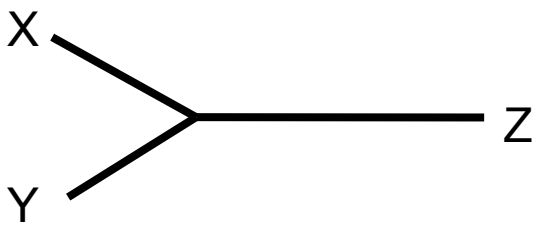


4-point condition is met

Ultrametric Tree
Stricter than Additive Tree
Efficient Solution

The ultrametric condition for metrics:

$$d(x,y) \leq \max(d(x,z), d(y,z))$$



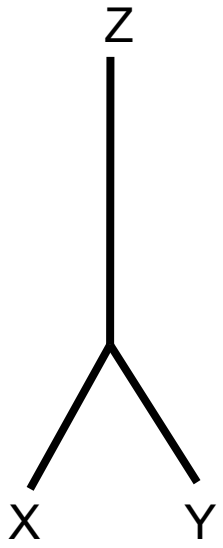
3-point condition: For any 3 points, the following holds

$$d(x,y) \leq d(x,z)$$

$$d(x,y) \leq d(y,z)$$

$$d(x,z) = d(y,z)$$

Ultrametric: Intuitive



The distance from Z to X is the same as the distance from Z to Y

More generally, the distances from leaves sharing any common ancestor are equal

Distances from root to leaves are strictly diminishing

Ultrametric Tree Test

- Build a distance matrix
- Then, $\forall i, j, k$

$$d(i,j)_{\max} = d(i,k)_{\max}$$

$$or$$

$$d(i,k)_{\max} = d(j,k)_{\max}$$

Ultrametric Tree

- Assumes evolution proceeds at the same rate along all edges (*i.e.* there is an evolutionary clock)
  - Point accepted mutation concept: Number of mutations proportional to time interval
- Is a natural topology for the heuristic UPGMA tree construction

Unweighted Pair Group Method with Arithmetic Mean

UPGMA

- Choose the pair X,Y with the smallest distance
- Combine X and Y into a singleton, XY
- The distance between any other point and the compound species XY is the arithmetic mean of the pairwise distances d_{ij} where i is a component species of X and j is a component species of Y.

Restated, the distance from k to xy is

$$(d(k,x)+d(k,y))/2$$

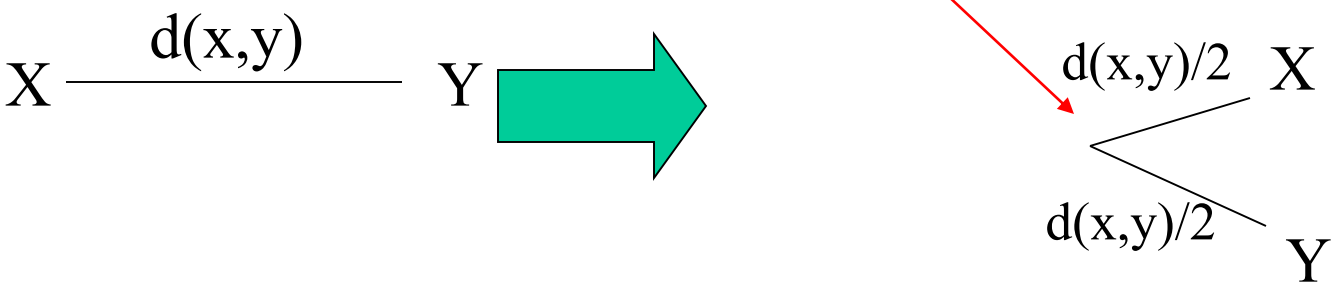
Unweighted Pair Group Method with Arithmetic Mean

UPGMA

- The matrix is now one order smaller
- Repeat the procedure until only two compound species remain.

UPGMA

Build an evolutionary tree such that from the point of branching between two (compound) species X and Y, the distance to each of their component species is the arithmetic mean, $d_{ij}/2$.



Unweighted Pair Group Method with Arithmetic Mean
Demonstration of UPGMA

Distance Matrix:

	A	B	C	D
A	0			
B	8	0		
C	7	9	0	
D	12	14	11	0

Closest pair

Notice that the 4-point condition holds in this matrix; it is additive

UPGMA

Points A and C will now become a complex, AC, to be regarded as its own ‘pseudo-point’

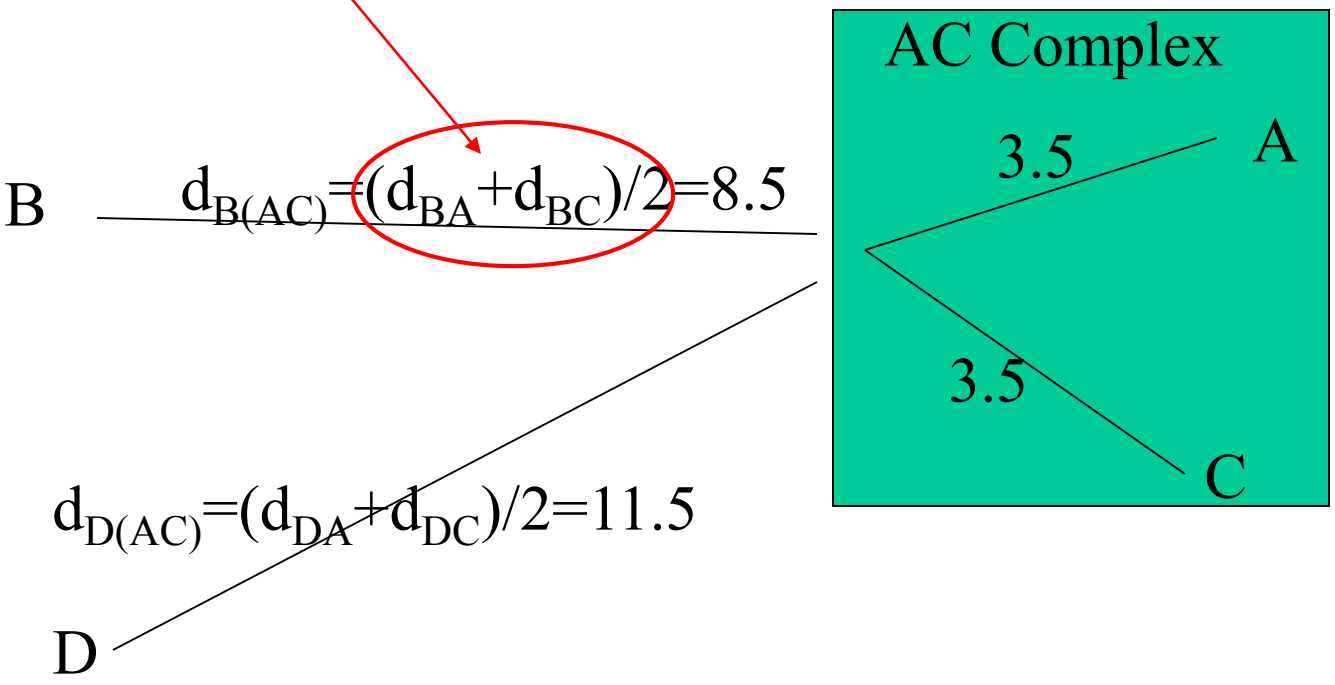
But we must find a distance from the other ‘real’ points to this complex

This distance is the mean of the distance from the remaining ‘real’ points to each of the points within the complex

UPGMA

Unweighted Mean

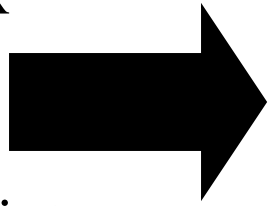
Choose closest pair: A and C and form a complex



Note that the even division of the AC complex is the ***ultrametric*** assumption

Re-represent the distance matrix

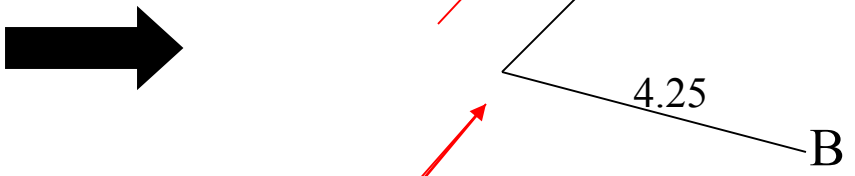
- It is one order smaller
- Use the complex as a single point

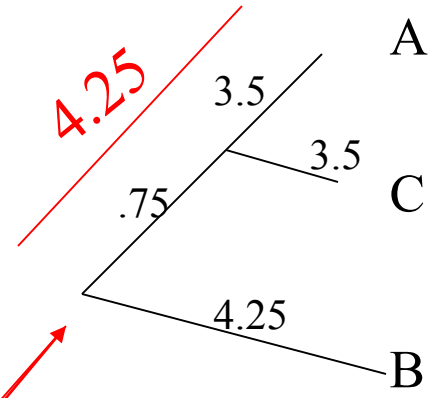


	(AC)	B	D
(AC)	0		
B	8.5	0	
D	11.5	14	0

Continue Refining the Distance Matrix

	(AC)	B	D
(AC)	0		
B	8.5	0	
D	11.5	14	0





Choose closest pair: (AC) and B

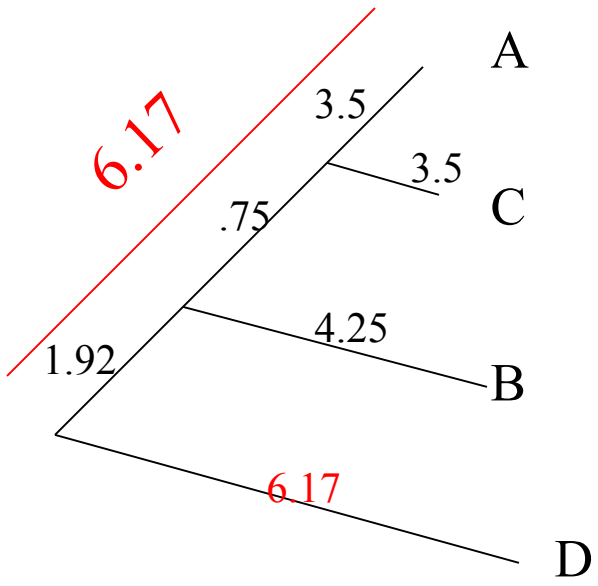
Branching node is at $[d_{B(AC)}]/2=4.25$

Next Refinement

$d_{(ABC)D} = (d_{AD}+d_{BD}+d_{CD})/3 = 12.33$

Add D with branching node at 12.33/2=6.17

	(ABC)	D
(ABC)	0	
D	12.33	0



Another Distance Method Neighbor-joining

- Distance Based
- Greedy Algorithm
- Looks like UPGMA but respects edge-lengths
  - No evolutionary clock

Strategy

The concept: Add branch nodes to the tree, and taxa to the branches, in such a way that the minimum total branch length is realized while, at the same time, each taxon ultimately becomes attached to a node.

The recipe:

- Taking the taxa pairwise, establish the total branch sum for each pair
- Find the ‘closest’ pair
- Create a new node between them
- Compute the distance of each taxon in this lowest-scoring pair to the new node
- Find the distance from all non-included taxa to this new node
- Create a new distance matrix from these distances
- Reiterate

When all free taxa are used up, the tree with the minimum total branch length will have been created

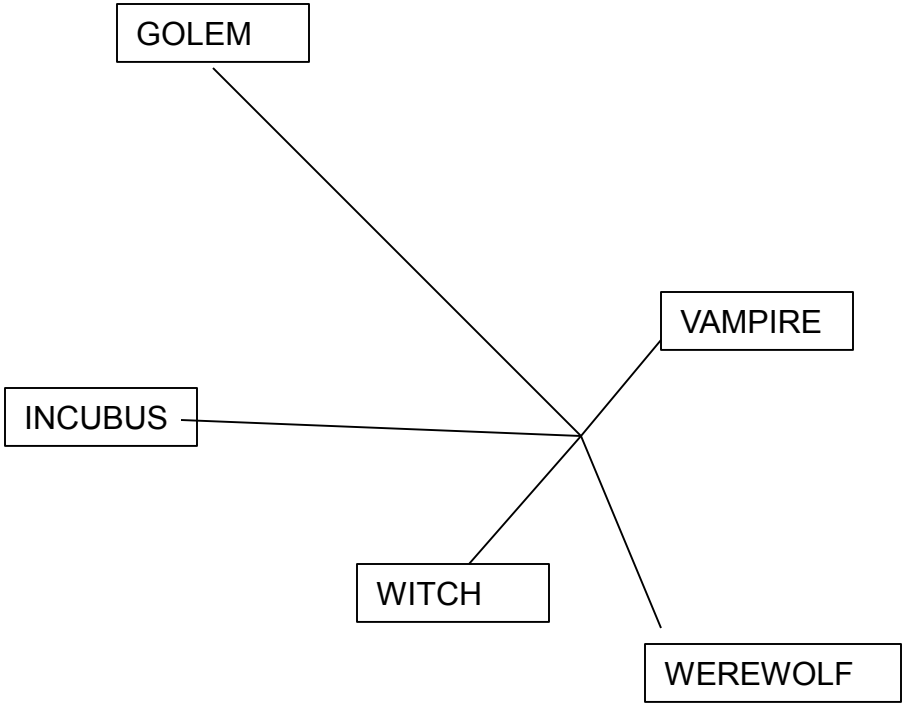
Example of Neighbor-joining

Consider 12 bp snippets from an important gene in these taxa:

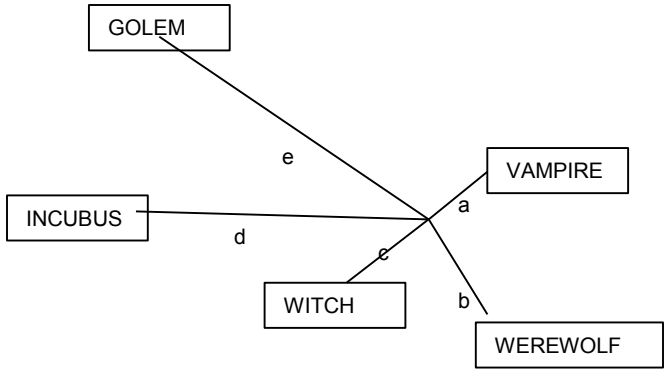
ACTTGACCTAAT Werewolf (ww)
AGTTGACCTAAT Vampire (v)
AGTTCACCTATT Witch (wt)
ACTACTGGATAT Incubus (i)
TGAACTGGATTA Golem (g)

	WW	*V*	*WT*	*I*	*G*
WW	0				
V	1	0			
WT	3	2	0		
I	7	7	8	0	
G	12	11	9	5	0

The sequences before any pairing and node incorporation



The first step is to establish the baseline branch length.



There are n(n-1)/2 pairwise distances between each of the n taxa, as represented in the distance matrix

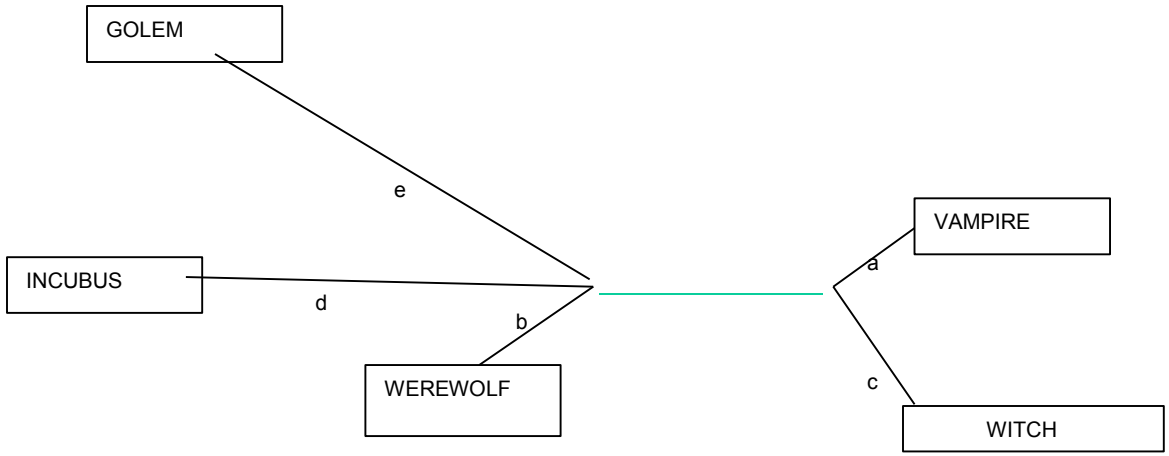
	WW	*V*	*WT*	*I*	*G*
WW	0				
V	1	0			
WT	3	2	0		
I	7	7	8	0	
G	12	11	9	5	0

The baseline branch length (path a +path b+ path c + path d+ path e) for this unpaired schema is simply the sum off all n(n-1)/2 distances between n taxa, divided by the number of times that a path has been traveled (n-1).

$$PathLength_{unpaired} = \sum \frac{d(i,j)}{n-1} \; \forall i, \forall j \quad j \neq i$$

So, in this example, the baseline path length is 65/4=16.25

The next step is to pair the two taxa whose pairing results in the shortest total path length. There is a different formula for the total path length when there are paired taxa.



All possible pairs (n(n-1)/2) must be tested. Here we <u>arbitrarily</u> start out with the vampire-witch pair, but we could have begun with any pair. Call the vampire X, the witch Y, and the vampire-witch distance d(X,Y)

The calculation of the total branch length now, when there are paired taxa, is:

Average distance of each element in the pair to each element not in the pair (eliminating multiply traveled paths)
+
Average of the paired taxa
+
Average of all distances not involving the pair (eliminating multiply traveled paths)

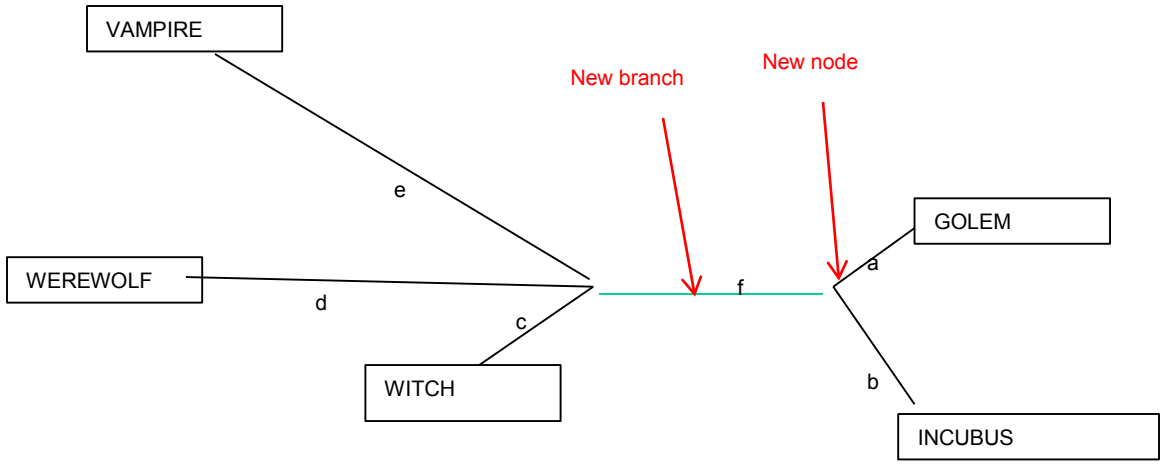
$$Branch\,Length_{based\ on\ pair\ X,Y} = \frac{\sum_{\substack{X,Y \in pair \\ \forall Z_{i,j} \notin pair}} [d(X,Z_i) + d(Y,Z_j)]}{2(n-2)} + \frac{d(X,Y)}{2} + \frac{\sum_{\forall Z_{i,j} \notin pair} d(Z_i,Z_j)}{n-2}$$

Perform this calculation based on every possible (n(n-1)/2) pairs of taxa, and find the one with the minimum total branch length.

In this example, the 10 possible total branch lengths were

14.8	17.2
15.7	17.5
16.8	17.5
17.7	16.3
15.5	13.5

In this example, the pair Golum-Incubus turns out to be better than Vampire-Witch, and is in fact, the minimum, with branch length of 13.5



With this pairing, a + b + c + d + e + f = 13.5

Next determine the distance from each member of the pair to the node (In UPGMA these distances are equal, under the ultrametric assumption, but that is not the case here as there is no evolutionary clock assumed). Restated, find *a* and *b* in the previous diagram, where *a* and *b* are not assumed to be equal

For 5 taxa in this example, where the optimal first pair is A (Incubus) and B (Golem),

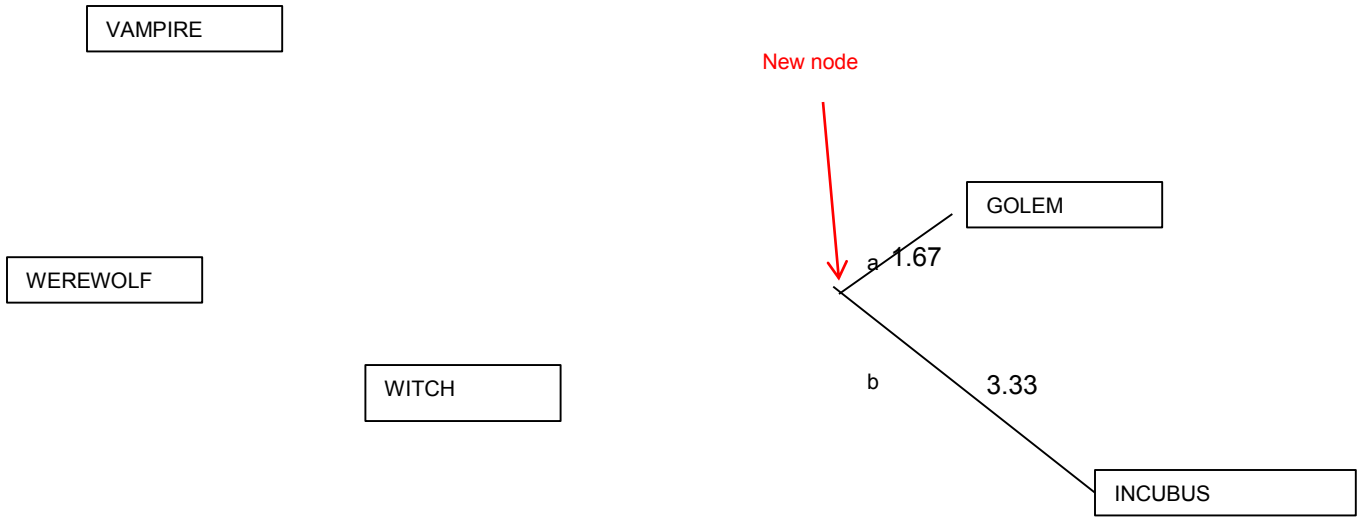
A to node = [d(A,B)+(d(A,C)+d(A,D)+d(A,E))/3-(d(B,C)+d(B,D)+d(B,E))/3]/2 =3.33

B to node = [d(B,A)+(d(B,C)+d(B,D)+d(B,E))/3-(d(A,C)+d(A,D)+d(A,E))/3]/2 = 1.67

or generally, for k taxa, Z_k not in the pair, and pair X,Y, the distance from X to the node is

$$length\, x = \frac{d(X,Y) + \frac{\sum_{Z_i \notin X,Y} d(X,Z_i) - \sum_{Z_i \notin X,Y} d(Y,Z_i)}{n-2}}{2}$$

So, the first pair is established as:



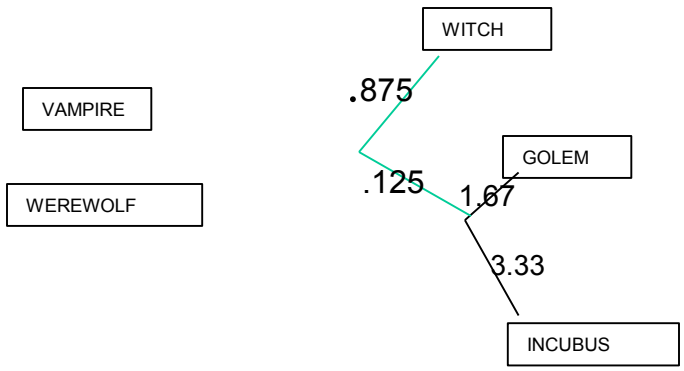
With this new pair, Golum-Incubus, a new distance matrix is formed, of reduced rank, using the average distance from each taxon to the pair.

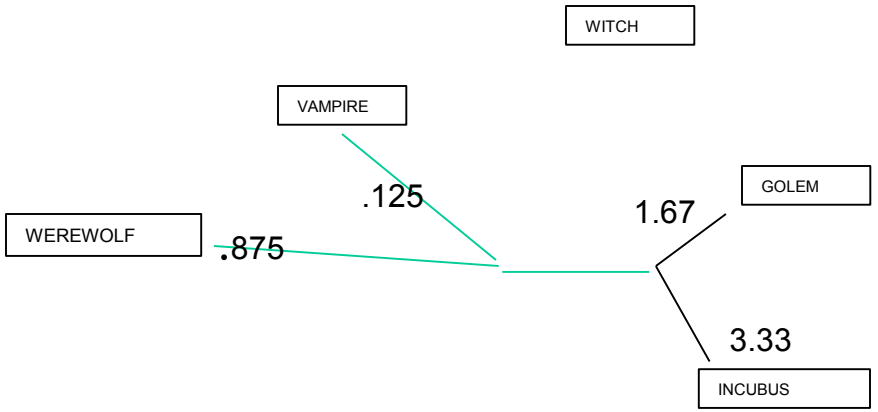
	Werewolf	*Vampire*	*Witch*	*Golum – Incubus*
Werewolf	0			
Vampire	1	0		
Witch	3	2	0	
Golum – Incubus	9.5	9	8.5	0

Again, the branch lengths of the possible pairs are calculated

10.62 11.13
11.25 11.25
11.13 10.62

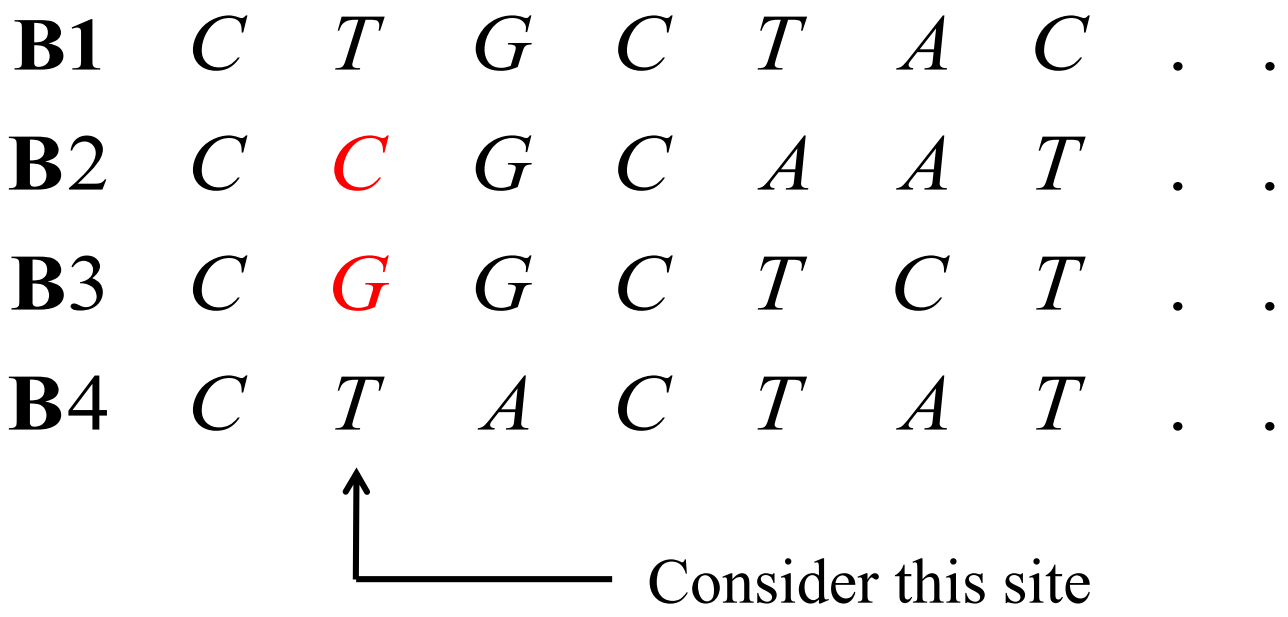
In this case either the Werewolf-Vampire pairing, or the Witch-(Golum-Incubus) pairing would be shortest, at 10.62 each





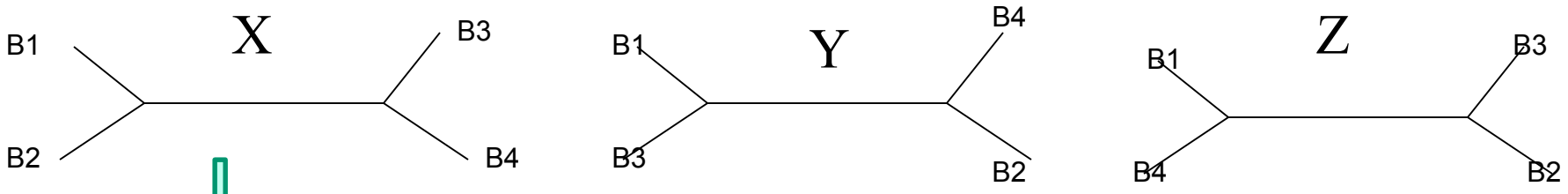
Maximum Likelihood Estimation Phylogeny

Begin with number of sequences that have been already aligned (here there are 4)

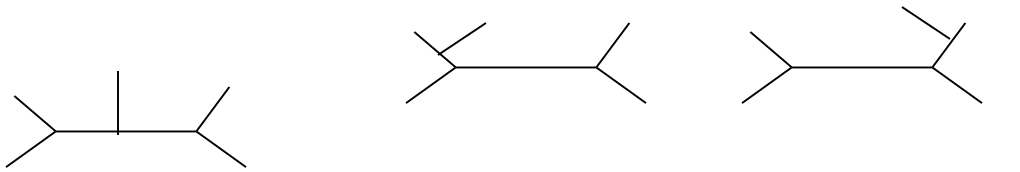


MLE

- For each site (*viz* column), consider all possible evolutions
- There are 3 unrooted trees and 5 rooted trees for each unrooted tree for just 4 sequence sites, or 15 rooted trees (Makes sense- recall $\frac{(2n-3)!}{2^{n-2}(n-2)!}$ Plug in n=4)

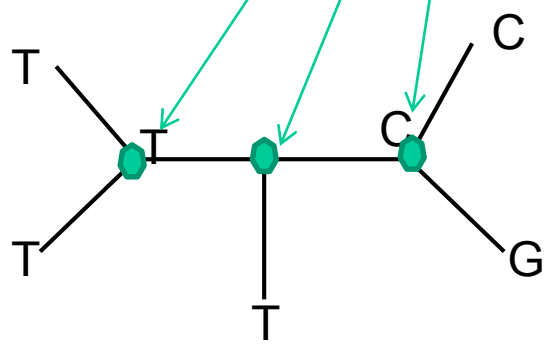


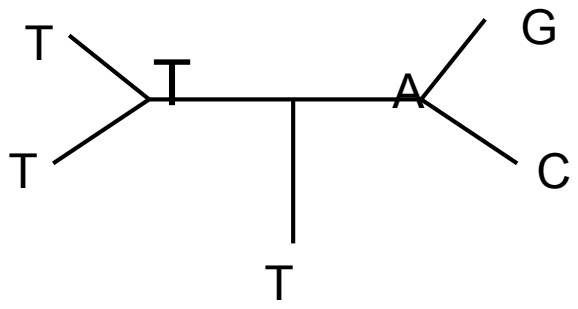
3 unrooted trees for 4 sequence sites

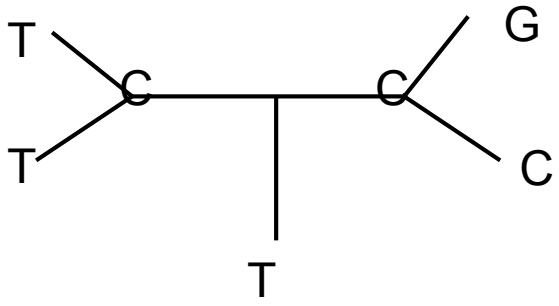


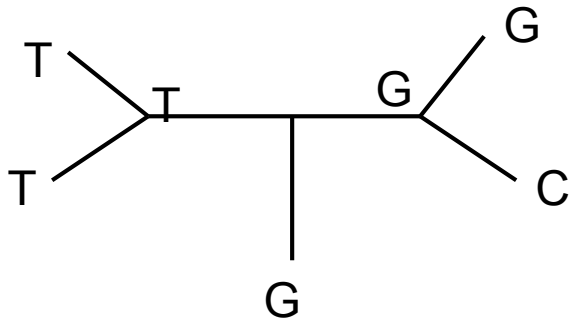
5 rooted trees for unrooted tree X

Let's start with this rooted tree, and see how we could evolve to the bases found at our MSA site of interest. Here are just 4 out of 64 possible ways (4 bases, 3 nodes=4^3). At first blush, Occam's Razor would favor top left as most probable, additional information about specific substitution probabilities withstanding.

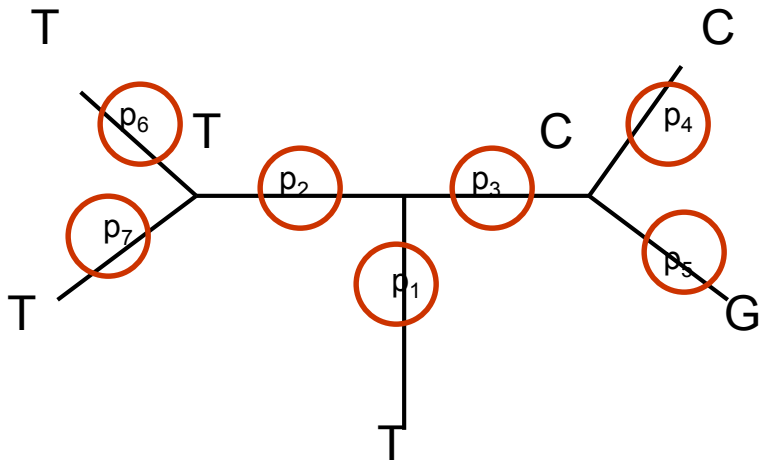








Now, for each change of base, there is an associated probability. This probability is derived from the evolutionary model, and would reflect unique probabilities for transitions, transversions, *etc*



The MLE method sums the product of these mutations, $p_1p_2\ p_3p_4\ p_5p_6\ p_7$ for each possible set of paths in a tree (64), then over each tree (15), and adds that to the sum for each site (column) in the sequence

Imagine the enormity of the calculation if there were more than 4 aligned sequences……..

But we are not done!!! ..

Using E-M, refine the most likely estimate for each edge length

..ugh