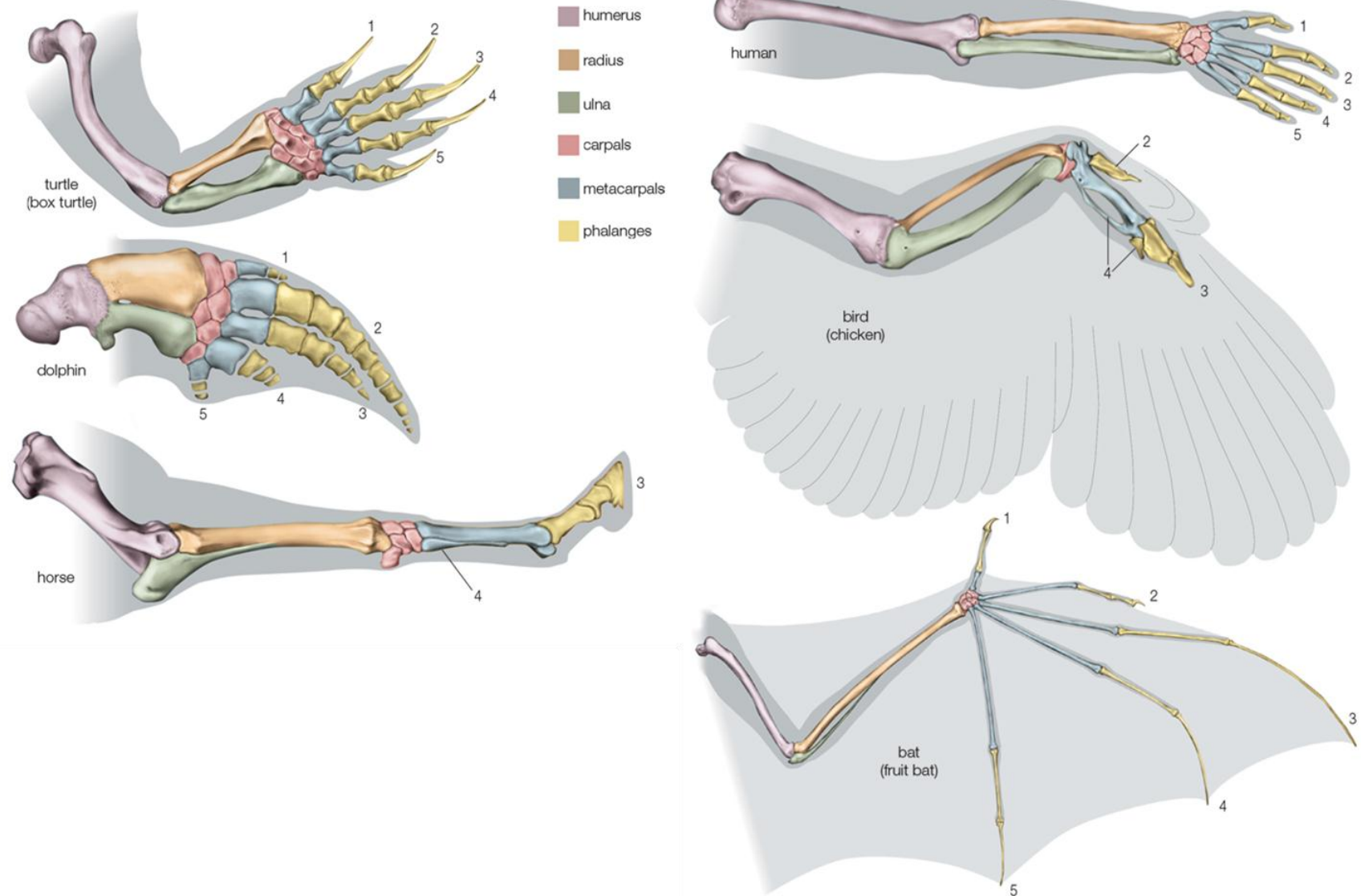


Building Phylogenetic Trees

Some slides adapted from a lecture by
Daniel Lopresti

A conventional view of evolution

Homologies of the forelimb in six vertebrates

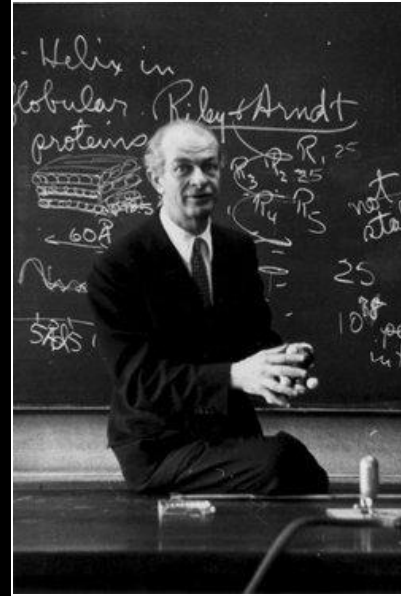


Origins of the study of Molecular Evolution

Emile
Zuckerkandl



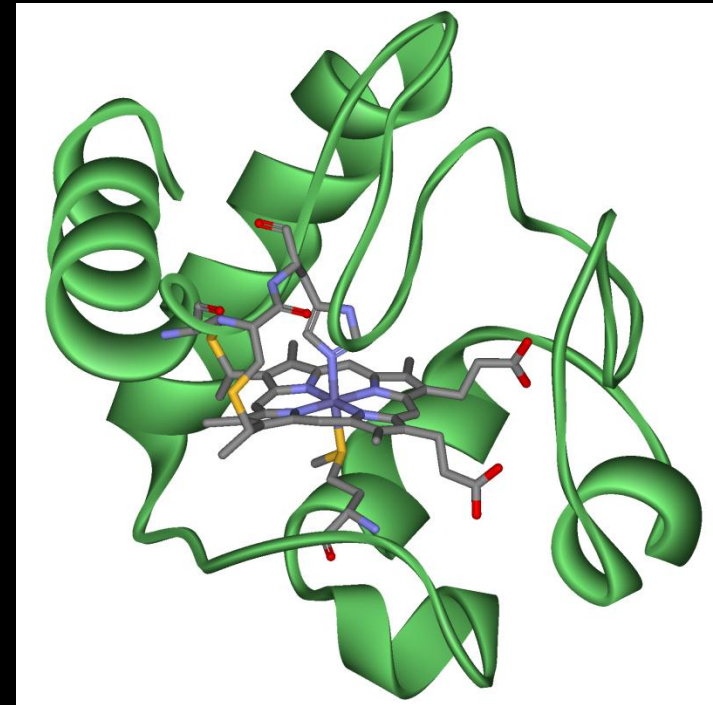
Linus
Pauling



- The Molecular Clock Hypothesis
 - Zuckerkandl and Pauling find that the rate of mutations in hemoglobin, in different species, seem consistent with fossil record
 - Thus, they predict when two organisms last shared a common ancestor, by counting mutations
 - At the time, this was outrageous

Support for the Molecular Clock Hypothesis

- Emmanuel Margoliash (Northwestern Univ):
 - In Cytochrome C from varying organisms, the number of mutations was proportional to the time since they last shared a common ancestor
- While this cannot be assessed for many organisms, differences between the cytochrome C of fish, frogs, chickens, rabbits, and horses varies at approximately 14%.
- Differences in the cytochrome C of bacteria and yeast with moths, tuna, and horses ranges between 64% and 69%.
- This was independent support for Zuckerkandl and Pauling's MCH



Cytochrome C, a pervasive electron transporter in all mitochondria

Why the MCH is so important

- Sequence alignment algorithms measure sequence identity
- If the number of variations – insertions, deletions, substitutions, etc – did not relate to evolutionary similarity, then sequence identity would be meaningless.
- The Molecular Clock Hypothesis is one of the foundations for using sequence identity as a measure of evolutionary distance
 - Otherwise there is no rationale for believing that two sequences are related or distantly related, based on sequence similarity

Issues with the Molecular Clock Hypothesis

- The Molecular Clock Hypothesis cannot be used as a general clock
 - It cannot substitute carbon dating for example
- Some genes substitute more often than others:

Rates of Amino Acid Substitution per 10^9 years in several proteins

Protein	Rate	Protein	Rate	Protein	Rate
Fibrinopeptides	9.0	Parathyrin	.73	Glucagon	.12
EGF	2.6	Trypsin	.59	Histone H2B	.09
Prolactin	1.7	Endorphin	.48	Ubiquitin	.01
Carbonic Anhydrase	1.6	Insulin	.44	Histon H4	.01

- The molecular clock hypothesis does not hold uniformly for proteins with different functions

Slow rates of substitution make sense here

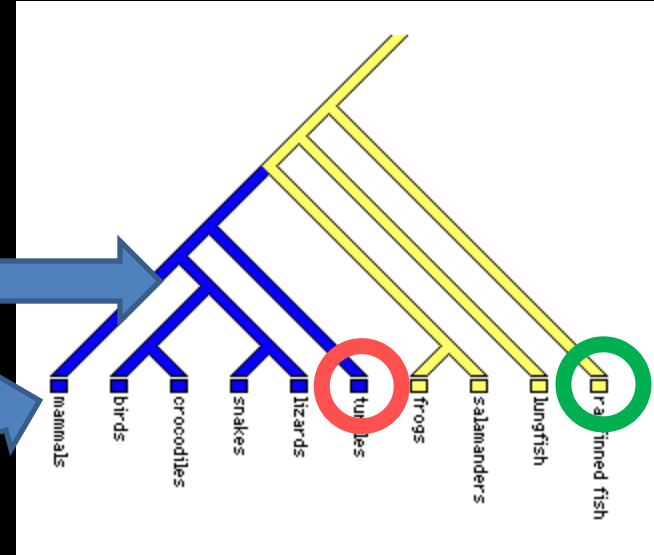
Rates of Amino Acid Substitution per 10^9 years in several proteins

Protein	Rate	Protein	Rate	Protein	Rate
Fibrinopeptides	9.0	Parathyrin	.73	Glucagon	.12
EGF	2.6	Trypsin	.59	Histone H2B	.09
Prolactin	1.7	Endorphin	.48	Ubiquitin	.01
Carbonic Anhydrase	1.6	Insulin	.44	Histone H4	.01

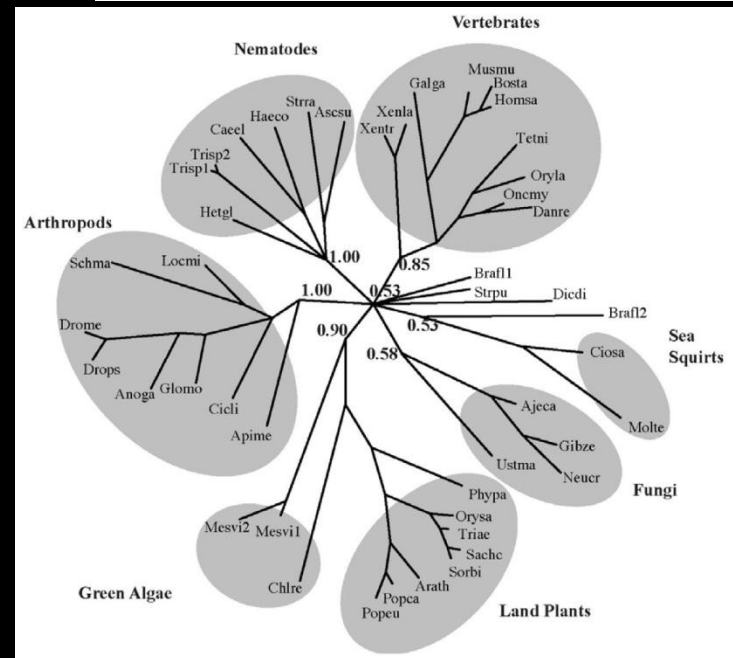
- **These proteins** play several critical roles
- Histones: wind and unwind DNA for storage.
 - Storage around histones is one way to stop expressing a gene – bad histones = bad regulation
- Glucagon: a hormone signal that leads to increased glucose in the blood
 - Improper binding lead to hyperglycemia; dangerous

Phylogenetic trees: a model of evolution

- A phylogenetic tree is a special kind of graph
 - Edges are called “branches”
 - Uniquely connect 2 nodes
 - The nodes are taxonomic units, or “taxa”

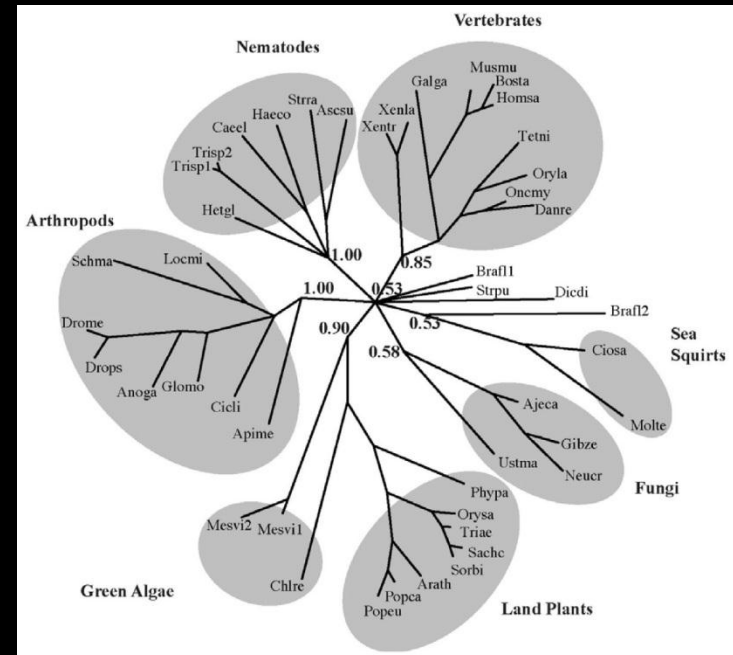
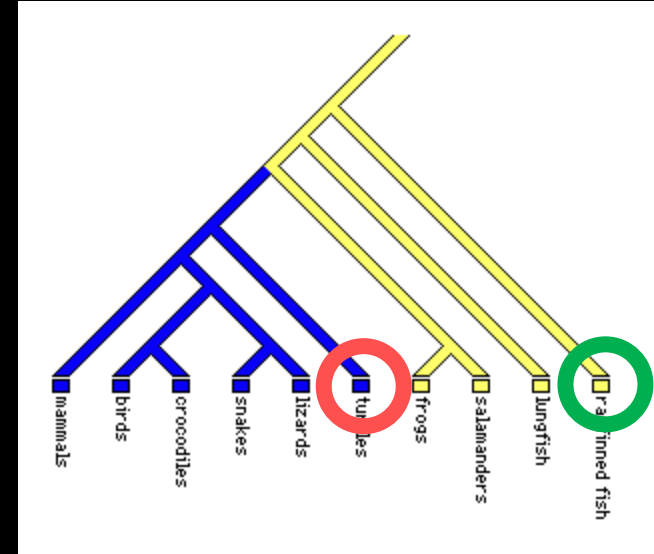


- **External nodes**, leaves, represent sequences
- **Internal nodes** represent hypothetical ancestors



Phylogenetic trees: Topology

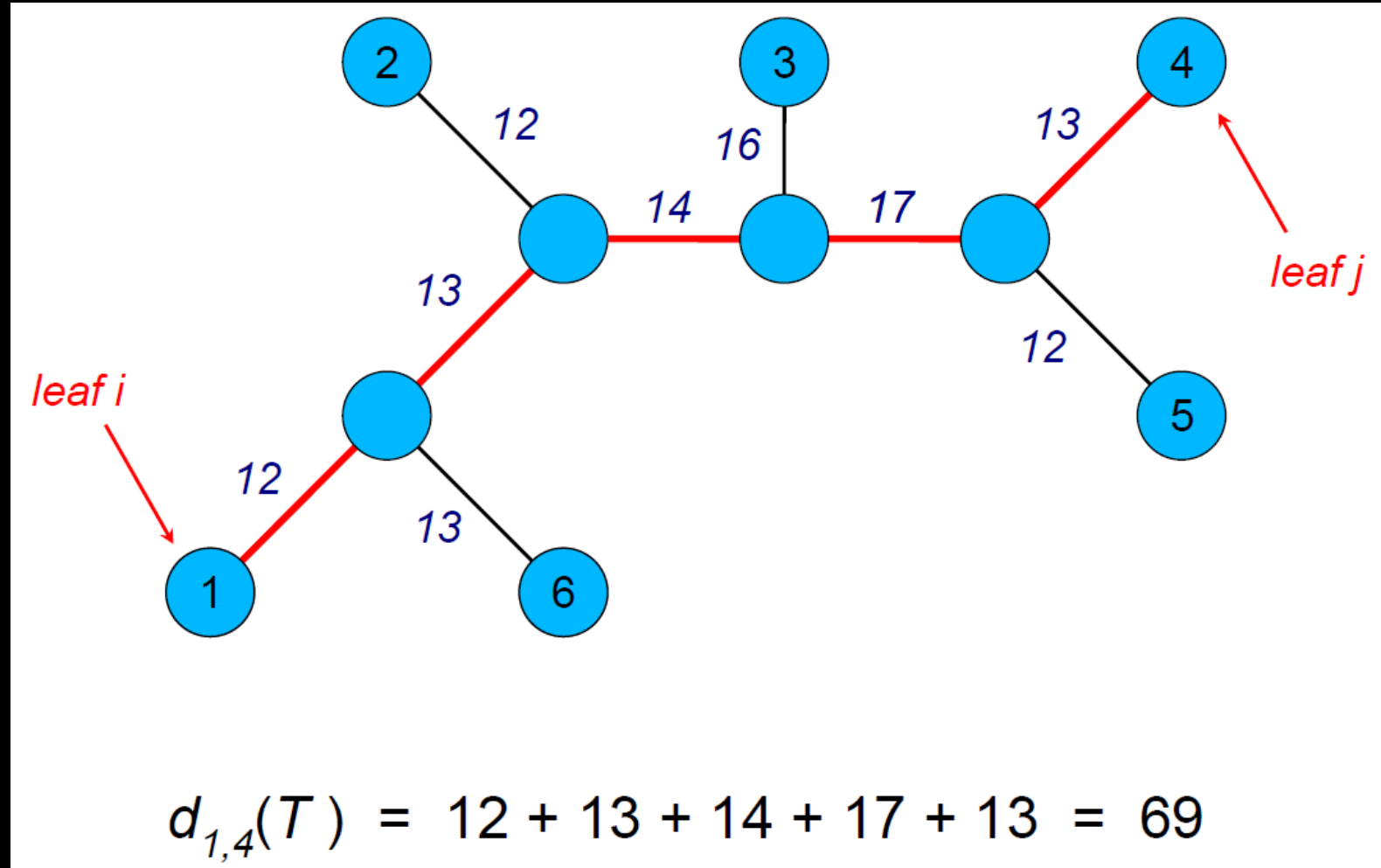
- The most significant aspect of the topology of a phylogenetic tree are
 - Branching
 - Branching illustrates the relationships between the taxa based on hypothetical ancestry
 - Branch lengths
 - Branch Lengths indicate the degree of ancestral similarity
 - This can even be proportional to sequence identity, # mutations
- A clade is all taxa that share a common ancestor, including the ancestor



Edge weights in Phylogenetic Trees

- Edges may have weights reflecting:
 - Number of mutations on evolutionary path from one species to another.
 - Time estimate for evolution of one species into another.
- In a tree T , we often compute:
- $d_{i,j}(T)$: the length of a path between leaves i and j
- $d_{i,j}(T)$ is called the *tree distance* between i and j .

Visualizing Tree Distance



- Here, walking on the tree between nodes *i* and *j* requires several edges, making the tree distance large relative to the size of the tree

The five stages of Phylogenetic analysis

- 1) Sequence Acquisition
- 2) Multiple sequence alignment
- 3) Models of DNA and amino acid substitution
- 4) Tree Construction
- 5) Tree Evaluation

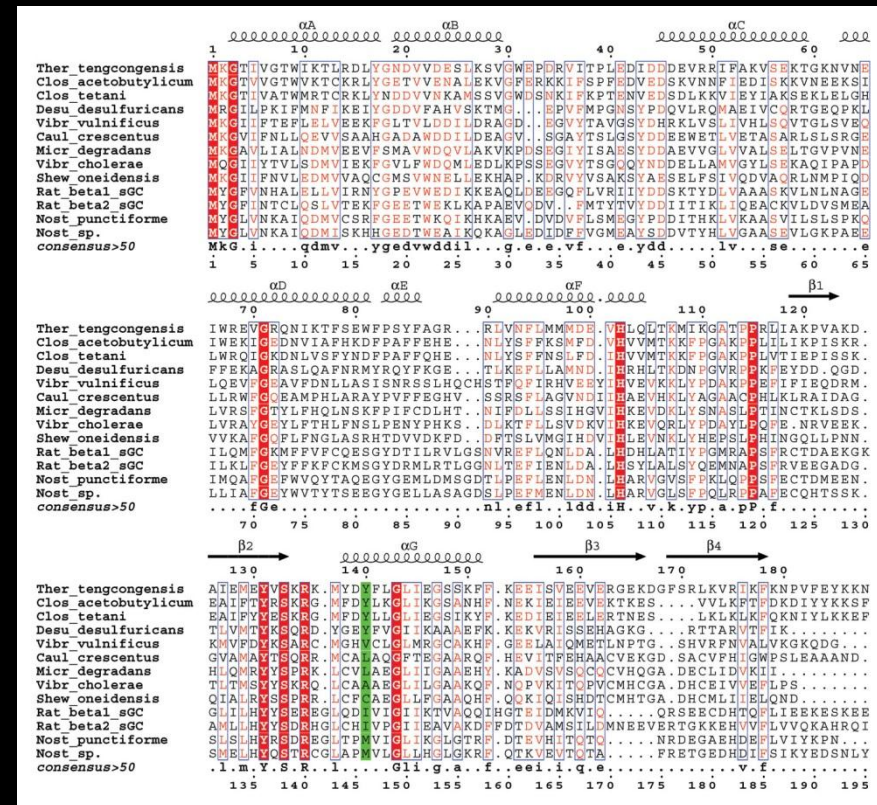
Step 1: Sequence Acquisition

- This step is simply getting the sequences
- Here, the most important aspect is diversity and balance.
- Diversity
 - If you want all examples of a protein found in all bacteria, select protein sequences from every subfamily of bacteria
- Balance
 - Avoid having more sequences of one subfamily than another subfamily
- BLAST and geneBank are excellent sources of this kind of sequence information

Step 2: Multiple Sequence Alignment

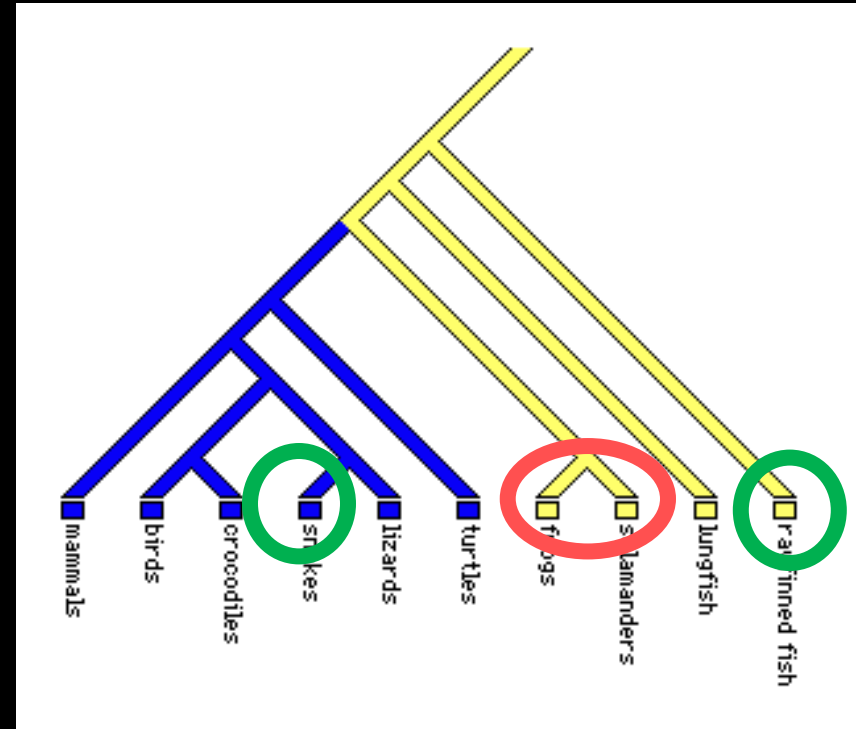
- ClustalW computes this alignment for you, but it is best to check to make sure the data is logical:

- 1) Inspect the alignment and make sure that the variations in these proteins is happening in similar places
- 2) Multiple sequence alignment algorithms might separate sequences that are too different. It may be necessary to modify gap opening/extension costs to get the proper alignment
- 3) Using only a portion of the sequence can help to get a proper alignment and not get thrown off



Step 3: Modeling DNA and substitution

- Phylogenetic trees are generated based on “distances” evaluated between difference sequences.
 - Sequences that have short distances between them are clustered in a **small clade**
 - Sequences that have **longer distances** are clustered in separate clades, if they are closer to other sequences
- We model DNA and substitution to compute these distances accurately



Step 3: Modeling DNA and substitution

- Before, when we scored nucleotide match and mismatch for sequence alignment, the purpose was to produce scores that cause dynamic programming to produce desirable alignments
- Here, we seek to score the quality of an alignment based on the substitutions indicated by the alignment, **so we can score distances between sequences**
- Scoring an alignment produces more sensitive measures of sequence variation than simply sequence identity

The Hamming Distance

CAGTA
CAT--

- The Hamming distance simply scores a normalized +1 for all similar aligned nucleotides, and -1 for all mismatches:

$$D = \frac{n}{N} \times 100$$

- The hamming distance is almost identical to sequence identity as a scoring distance, normalized to the total number of nucleotides

The Poisson correction

- The Hamming distance penalizes very heavily for differences in nucleotides that have differences
- The Poisson correction assumes that nucleotides have a uniform rate of variation with equal probabilities

$$P = -\ln(1 - p)$$

- Where P is the Poisson distance, and p is the number of residues that differ

How these distances are used

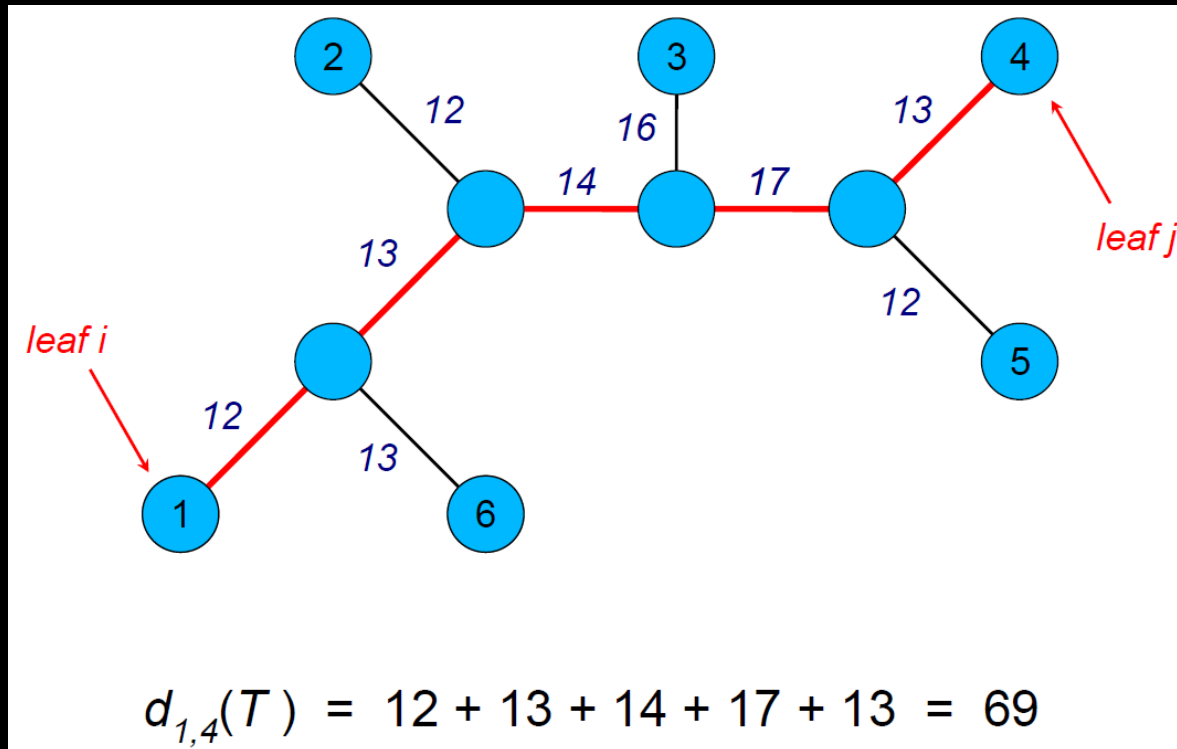
- Distances computed with the Poisson correction or other scoring functions are put into the distance matrix.
- The distance matrix is used for the next stage:
Tree Building

Step 4: Building Phylogenetic Trees

- Given n species, we know we can compute an $n \times n$ distance matrix $D_{i,j}$.
- Evolution of these genes is described by a tree that we do not know
- We need an algorithm to construct a tree that best fits distance matrix $D_{i,j}$.
- In particular, fitting means finding a tree T such that:
 - $D_{i,j} = d_{i,j}(T)$ for all genes i and j

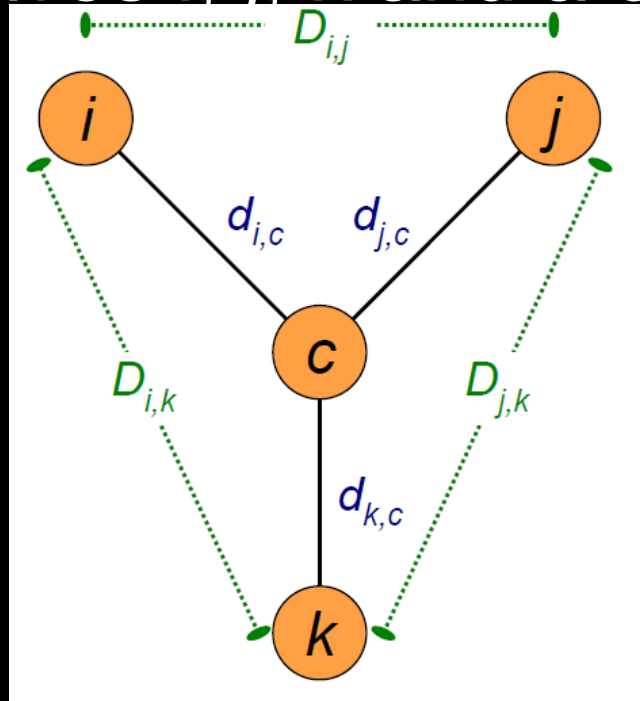
Finding a tree that fits

- In particular, this means that a tree that fits the matrix $D_{i,j}$ will have edge lengths between all pairs of nodes that add up to distance in the matrix.

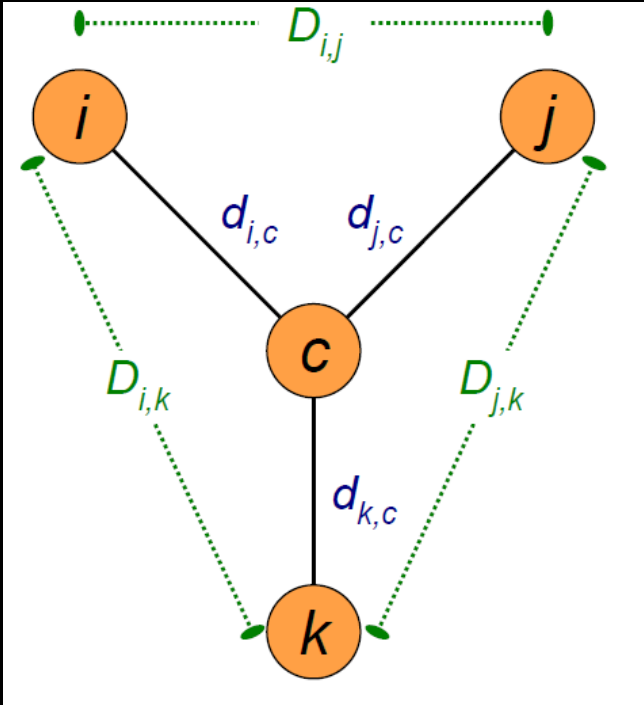


How can we find a tree that works?

- Lets look at a simple example: the case of three nodes.
- Tree reconstruction for any 3x3 matrix is straightforward.
- We have 3 leaves i, j, k and a center vertex c .



First, an observation



Observe:

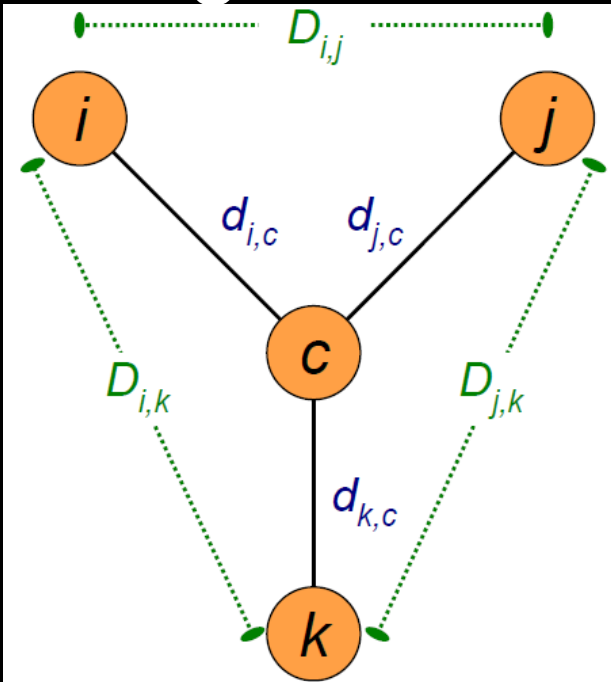
$$d_{i,c} + d_{j,c} = D_{i,j}$$

$$d_{i,c} + d_{k,c} = D_{i,k}$$

$$d_{j,c} + d_{k,c} = D_{j,k}$$

- Using these three simultaneous equations, we can solve for the optimal tree in this case

Solving for a simple optimal tree



Observe:

$$d_{i,c} + d_{j,c} = D_{i,j}$$

$$d_{i,c} + d_{k,c} = D_{i,k}$$

$$d_{j,c} + d_{k,c} = D_{j,k}$$

- Using these three simultaneous equations, we can solve for the optimal tree in this case:

Observe:

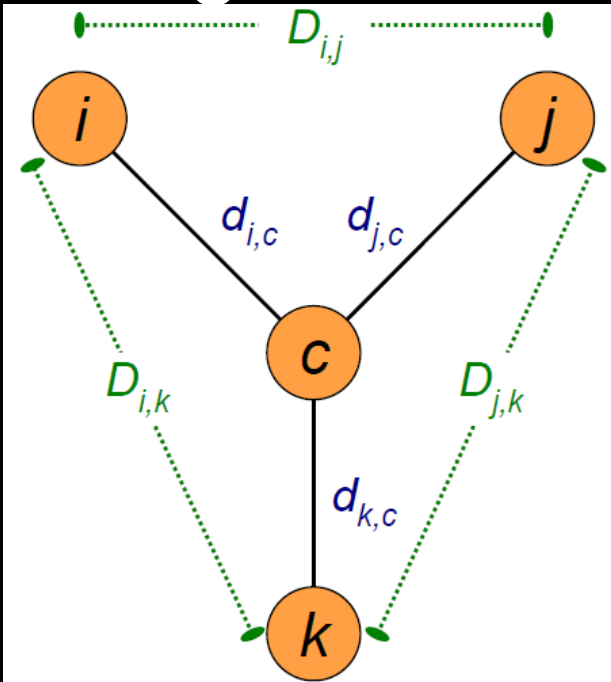
$$\begin{array}{rcl} d_{i,c} + d_{j,c} & = & D_{i,j} \\ + d_{i,c} + d_{k,c} & = & D_{i,k} \\ \hline 2d_{i,c} + d_{j,c} + d_{k,c} & = & D_{i,j} + D_{i,k} \end{array}$$

$= D_{j,k}$

$$2d_{i,c} + D_{j,k} = D_{i,j} + D_{i,k}$$

so $d_{i,c} = (D_{i,j} + D_{i,k} - D_{j,k}) / 2$

Solving for a simple optimal tree



Observe:

$$d_{i,c} + d_{j,c} = D_{i,j}$$

$$d_{i,c} + d_{k,c} = D_{i,k}$$

$$d_{j,c} + d_{k,c} = D_{j,k}$$

- The same is true for the other two edges:

Observe:

$$\begin{array}{rcl} d_{i,c} + d_{j,c} & = & D_{i,j} \\ + d_{i,c} + d_{k,c} & = & D_{i,k} \\ \hline 2d_{i,c} + d_{j,c} + d_{k,c} & = & D_{i,j} + D_{i,k} \end{array}$$

$= D_{j,k}$

$$\begin{array}{l} 2d_{i,c} + D_{j,k} = D_{i,j} + D_{i,k} \\ \text{so } d_{i,c} = (D_{i,j} + D_{i,k} - D_{j,k}) / 2 \end{array}$$

$$d_{j,c} = (D_{i,j} + D_{j,k} - D_{i,k}) / 2$$

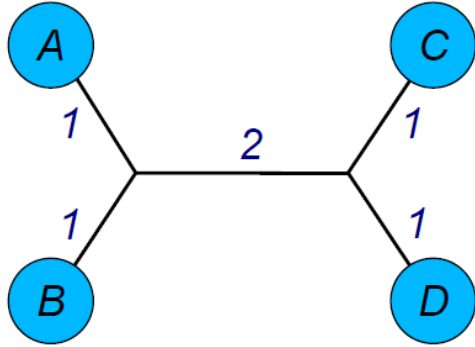

$$d_{k,c} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

Solving for more general trees

- It's clear that we have a method for solving a three node tree. However, this approach likely does not generalize so easily for larger trees
- A tree with n leaves has $2n-3$ edges.
- This means fitting a given tree to a distance matrix D requires solving a system of n choose 2 equations with $2n-3$ variables
- This is not always possible to solve for $n > 3$.

When do we have a solution?

- We call a matrix *additive* if there exists a tree T with $d_{i,j}(T) = D_{i,j}$

<i>Additive</i>		A	B	C	D
	A	0	2	4	4
	B	2	0	4	4
	C	4	4	0	2
	D	4	4	2	0
					
<i>Non-additive</i>		A	B	C	D
	A	0	2	2	2
	B	2	0	3	2
	C	2	3	0	2
	D	2	2	2	0
					

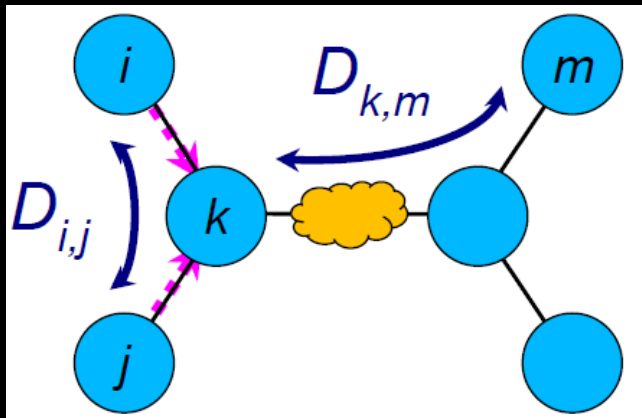
The Distance-Based Phylogeny Problem

- Reconstruct an evolutionary tree from a distance matrix.
- **Input:** An $n \times n$ distance matrix $D_{i,j}$.
- **Output:** A weighted tree T with n leaves fitting $D_{i,j}$.

If $D_{i,j}$ is additive, this problem has a solution and there is a simple algorithm to solve it.

One solution: neighbor joining

- Genes that are similar ought to be in physical proximity when we build our tree. Close genes can be combined in an iterative process that adds structure and reduces size of problem in each step.
- Find neighboring leaves (genes) i and j with parent k .
- Remove rows and columns of i and j from distance matrix.
- Add a new row and column corresponding to k , where distance from k to any other leaf m can be computed as:



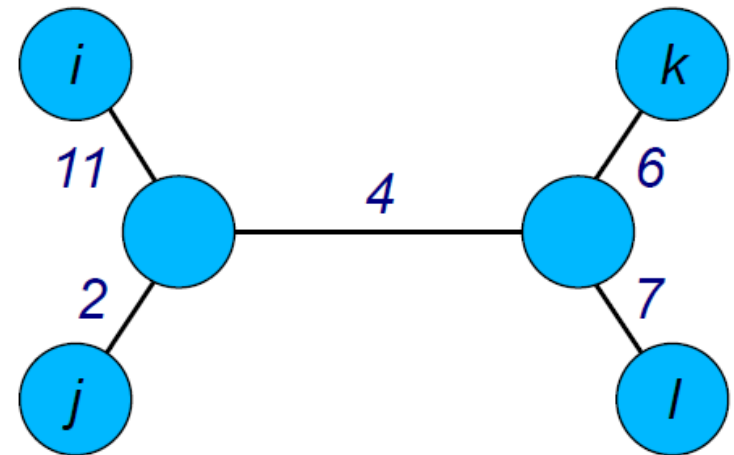
$$D_{k,m} = (D_{i,m} + D_{j,m} - D_{i,j}) / 2$$

How do we find neighbors?

- To find neighboring leaves, can we simply select pair of leaves that are closest?
- Actually no: closest leaves are not necessarily siblings.

Here, i and j are siblings and j and k are not, but $d_{i,j} = 13 > d_{j,k} = 12$:

So we would next combine j and k , which is wrong



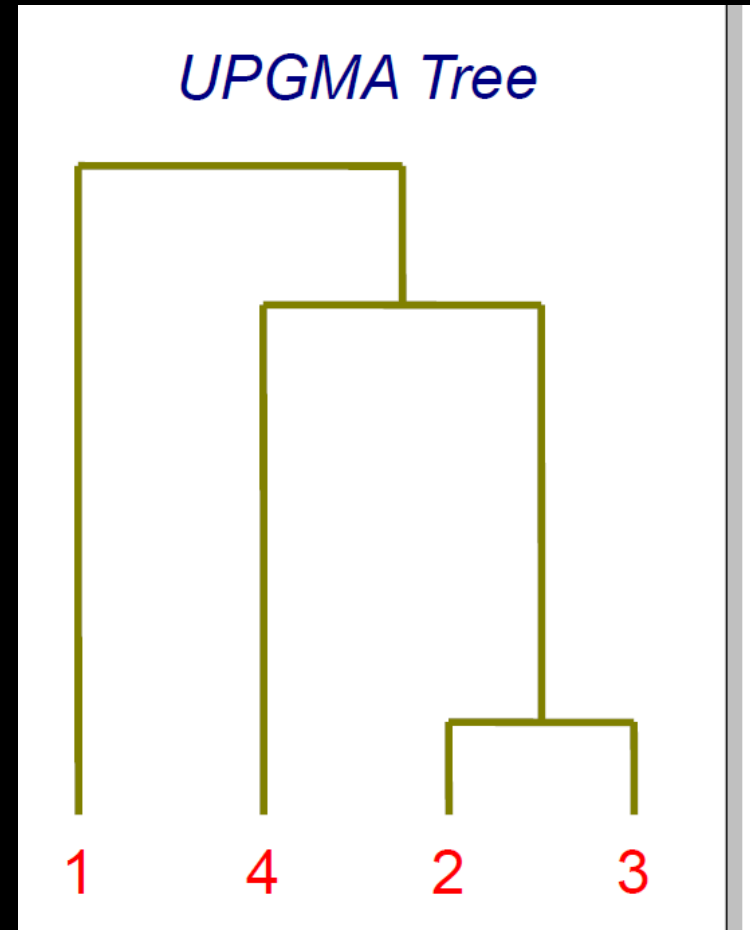
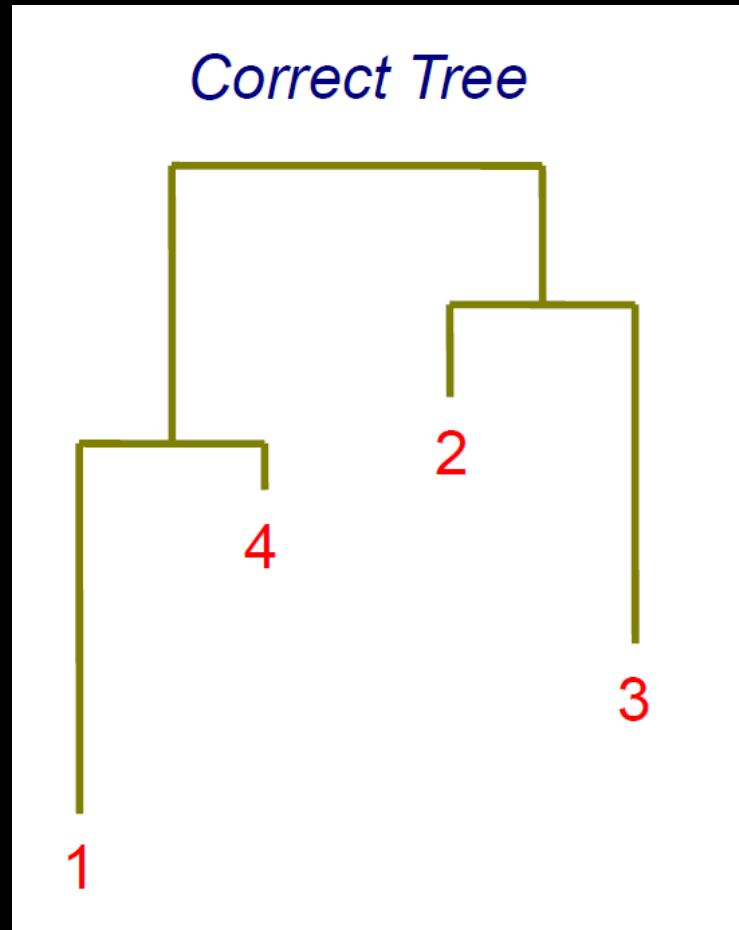
- In 1987, Naruya Saitou and Masatoshi Nei developed a neighbor joining algorithm for phylogenetic tree reconstruction.
- Finds a pair of leaves that are close to each other but far from other leaves: implicitly finds neighboring leaves.
- Advantages: works well for additive and other distance matrices – it does not have “flawed molecular clock” assumption (all leaves same distance from root).

UPGMA trees

- UPGMA (Unweighted Pair Group Method with Arithmetic Mean) is a clustering algorithm that:
- Computes distance between clusters using average pairwise distance.
- Assigns a height to every vertex in tree, effectively assuming presence of a molecular clock which “dates” vertex.

- UPGMA's weakness:
- Algorithm produces an *ultrametric tree*: distance from root to any leaf is same (i.e., assumes constant molecular clock; all species represented by leaves evolved at same rate.)

Visible Differences in UPGMA



- UPGMA assumes that all genes evolve at the same rate, and hence have the same distance from the root

Implementing UPGMA

UPGMA(D)

- Assign each x_i to its own cluster C_i
- Define one leaf per sequence, each at height 0
- While (more than one cluster remains)
 - Find two clusters C_i & C_j such that $d_{i,j}$ is minimized
 - Let $C_k = C_i \cup C_j$
 - Add vertex between C_i and C_j to T at height $(d_{i,j} / 2)$
 - Delete C_i and C_j
- Return T

Clustering in UPGMA

- Given two disjoint clusters C_i , C_j of sequences, we define:

$$d_{i,j} \equiv \frac{1}{|C_i| \times |C_j|} \sum_{\{p \in C_i, q \in C_j\}} d_{p,q}$$

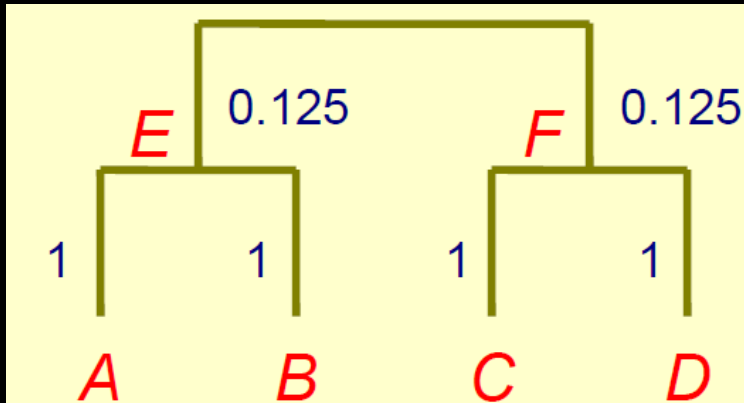
- Note that if $C_k = C_i \cup C_j$, then distance to another cluster C_l is:

$$d_{k,l} = \frac{d_{i,l}|C_i| + d_{j,l}|C_j|}{|C_i| + |C_j|}$$

UPGMA on a non-additive matrix

This is an example of a non-additive matrix

	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0



UPGMA step 1:
Merge A, B, to E

	E	C	D
E	0	2.5	2
C	2.5	0	2
D	2	2	0

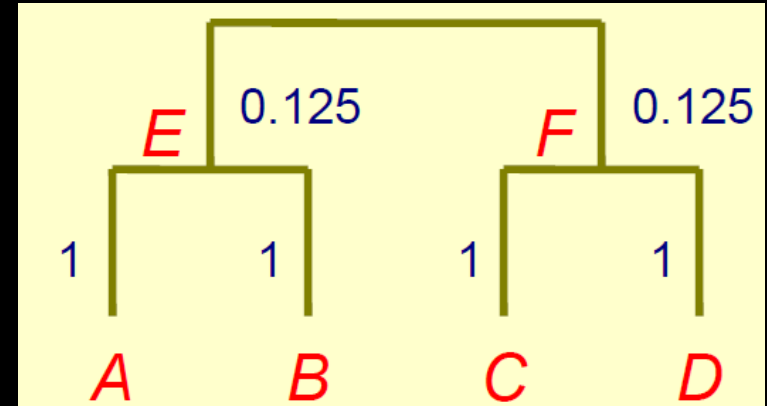
UPGMA step 2
Merge C,D, to F

	E	F
E	0	2.25
F	2.25	0

UPGMA step 3
Merge E,F

UPGMA approximates non-additive matrices

- This is the tree computed by UPGMA



Original Distance Matrix

	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

Implied UPGMA matrix

	A	B	C	D
A	0	2	2.5	2.5
B	2	0	2.5	2.5
C	2.5	2.5	0	2
D	2.5	2.5	2	0

Wrap up

- UPGMA gets around the problem of finding a tree for a non-additive matrix by approximating the tree
- UPGMA loses some accuracy because it assumes that all members of the tree evolve at the same rate

Questions