Unweighted Pair-Group Method with Arithmetic Mean

↑

Reminder: UPGMA Algo

Input: $S_1 \dots S_n$

Output: Tree T with branch lengths

① Estimate distance matrix D from pairwise alignments

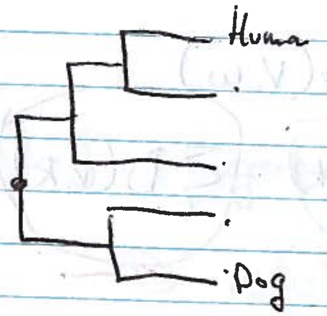
② Repeat

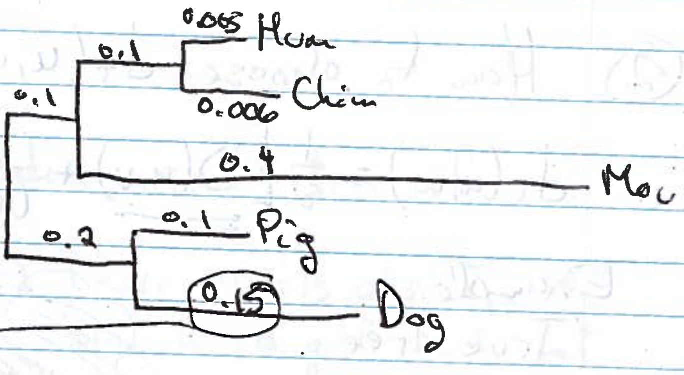
```
    2.1. Choose two nodes (u,v) to pair up
    2.2. Remove (u,v) from D
    2.3. Create new node w as ancestor of u,v
    2.4. Add new row/column for w in D.
```

UPGMA produces correct output if

① Distances are estimated perfectly accurately

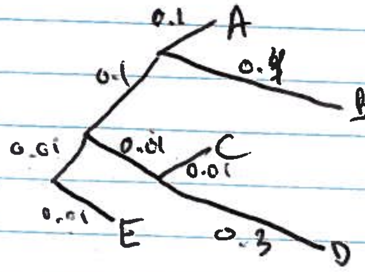
② Mutation rate is constant



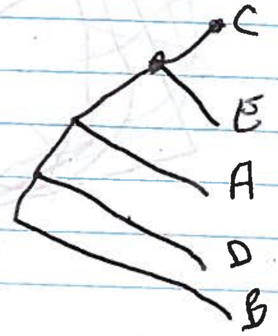


→ Expected number of subst. per site

Suppose true tree



UPGMA



Wrong Tree

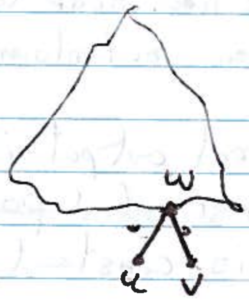
Neighbor-joining Algo (Nei, Saitou)

Same as UPGMA, but different $\overset{\text{node}}{\vee}$ pair selection rule

① Calculate $Q_{n\times n}$, where $Q(i,j) = \sum_{k=1}^{n} D(i,k) + \sum_{k=1}^{n} D(j,k) - (n-2)D($

Theorem: $u,v \leftarrow \underset{i,j}{\arg\max}\{Q(i,j)\}$, then

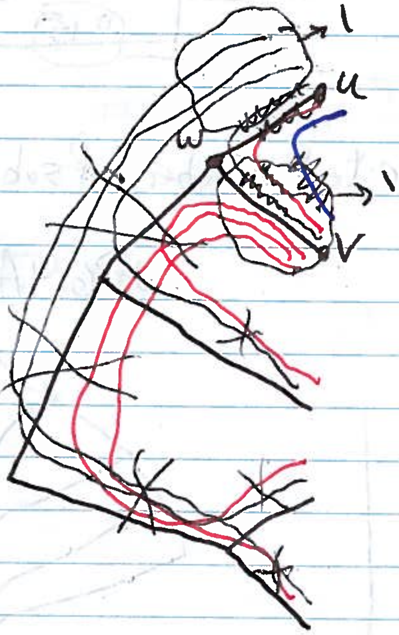
nodes u and v must be a cherry



② How to choose $d_T(u,w)$ and $d_T(v,w)$

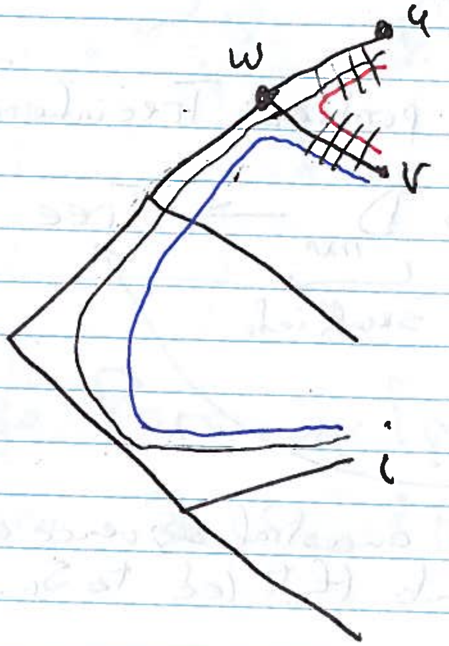
$$d_T(u,w) = \frac{1}{2}\left[\underline{D(u,v)} + \frac{1}{(n-2)}\left(\boxed{\sum_k D(u,k)} - \boxed{\sum_k D(v,k)}\right)\right]$$

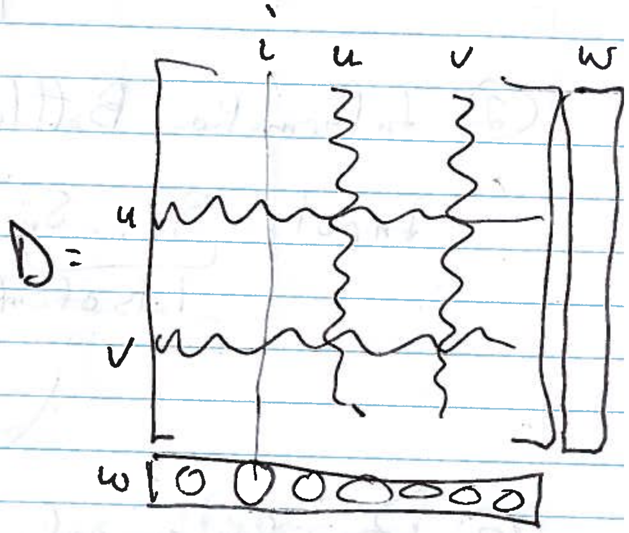
Example:
True tree



Adding row/col for w in D

$$D(w,i) = \frac{1}{2}\left(\underline{D(u,i)} + \underline{D(v,i)} - \underline{D(u,v)}\right)$$





N-J is garanteed to produce correct tree if D is ultrametric → There exists a tree T with branch length such that $d_T(i,j) = D(i,j)$

Running time: $O(n^4)$, but can be reduced to $O(n^3)$

Problems with N-J algo

① If D is not ultrametric (D is not estimated ~~with~~ perfecty)

$\Rightarrow$ N-J comes NO with garantees about optimality

② Information Bottleneck problem: Tree inferred may be inaccurate

$$\begin{tikzcd}
\text{Input: } \underbrace{S_1, \dots S_n}_{\text{lots of info.}} \arrow[r] \arrow[rr, bend right, "?"'] & \underbrace{D_{n \times n}}_{\text{small info}} \arrow[r] & \text{Tree}
\end{tikzcd}$$

③ Provide no info about ancestral sequence or about evolutionary events that led to $S_1 \dots S_n$