# Distance-based methods for construction of phylogenetic trees

## Lecture 6.4

*by Marina Barsky*

# Defining distance

- How to measure distance between 2 DNA molecules so it reflects the time since they have evolved from a common ancestor?

# The relative distance between genomes can be measured as the number of mutational events

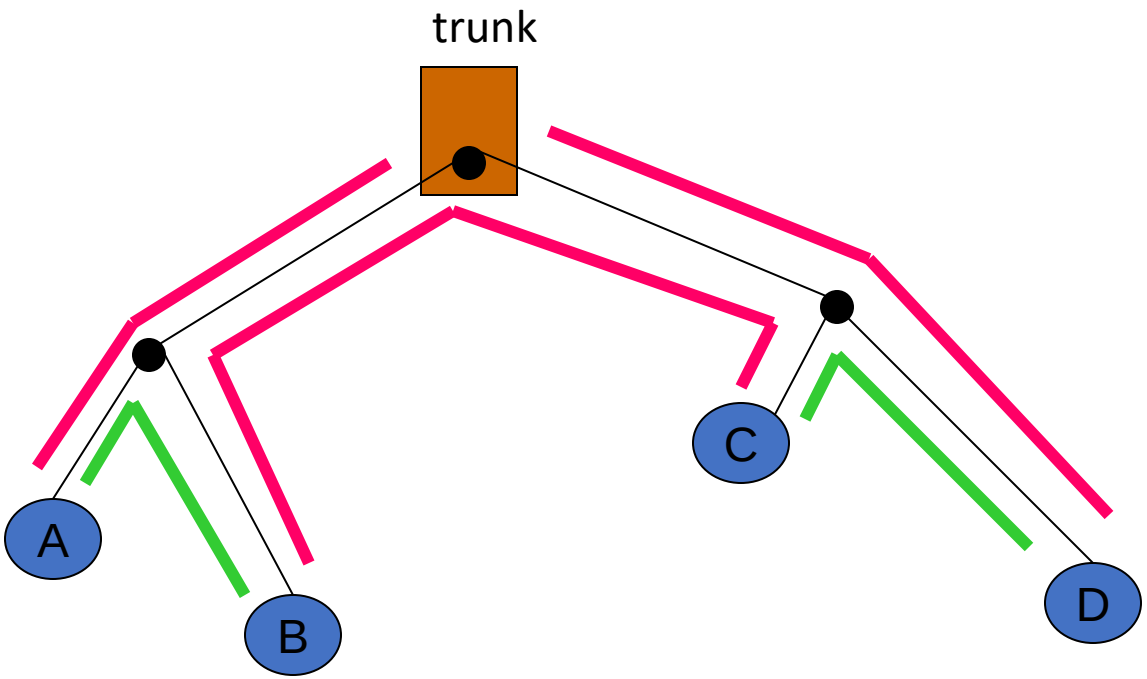
- Non computational: melting temperature of DNA hybrids
- Computational
  - Based on DNA or protein sequences
    - *Edit distance* – based on point mutations
  - Gene-sequence based
    - Alignment traces – align chromosomes from the different species, connect homologous genes by an edge. The number of crosses can be used as an evolutionary distance
- Better: combine different metrics

# Building the tree

- Given a set of pairwise distances, find the best tree for a given data

# Additive distances

- Distances that fit onto some tree are called *additive*. To determine if the distances are additive we use *the four point criterion:*

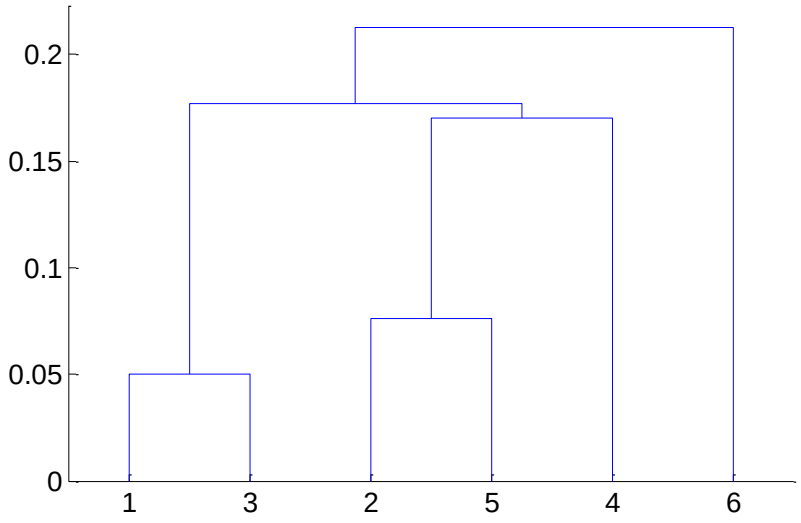


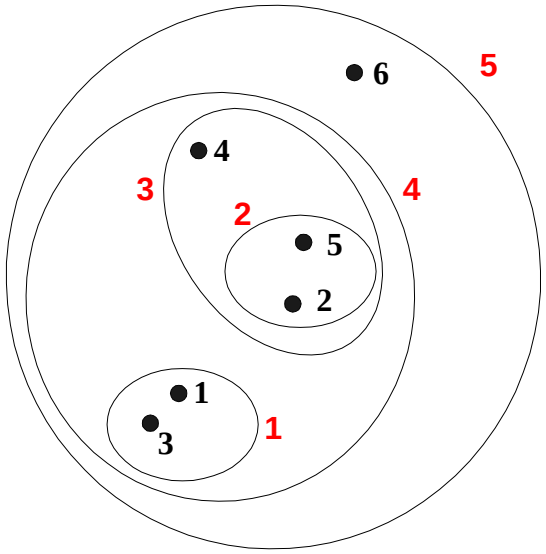

1. The sums of pairwise distances that traverse the trunk are equal
2. The sum of distances that traverse the trunk is >= the sum of remaining distances

$$d_{AD}+d_{BC}=d_{BD}+d_{AC}>=d_{AB}+d_{CD}$$

# Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree-like diagram that records the sequences of merges or splits



# Hierarchical Clustering

- Start with the points as individual clusters.
- At each step, merge the closest pair of clusters until only one cluster left.

**Algorithm**

Let each data point be a cluster

Compute the distance matrix

**Repeat**

Merge the two closest clusters

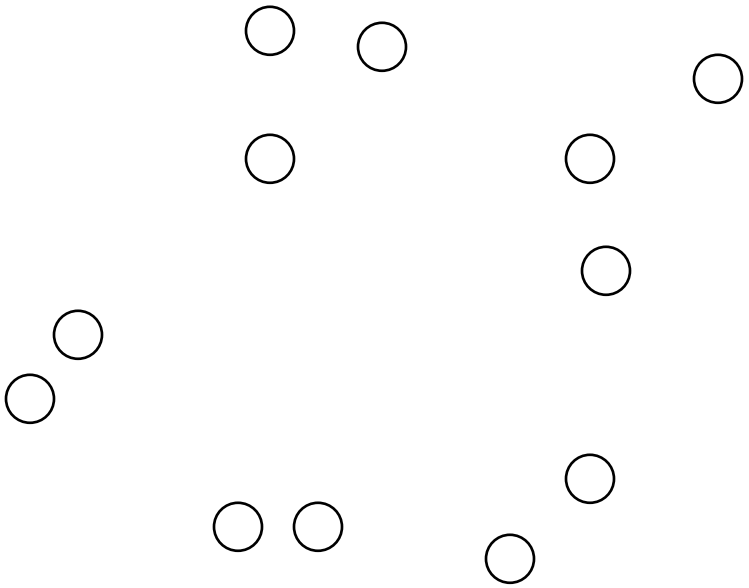
Update the distance matrix

**Until** only a single cluster remains

- Key operation is *the computation of the distance between two clusters.*

# Starting Situation

- Start with clusters of individual points and a proximity matrix

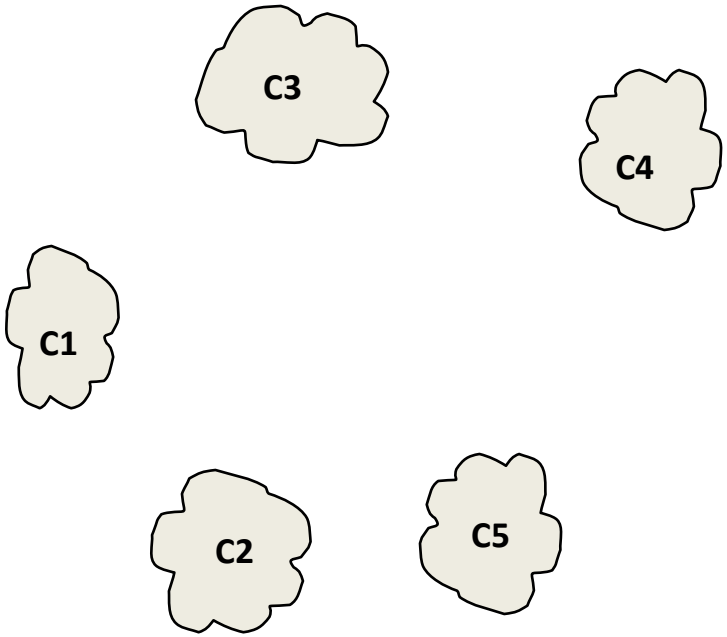


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

**Distance Matrix**

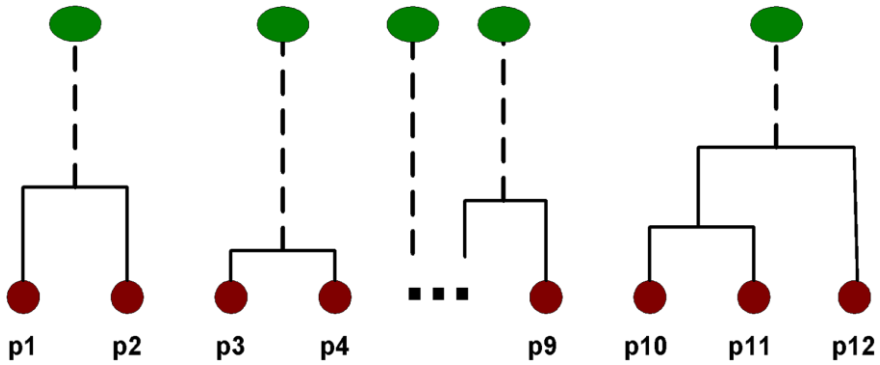
# Intermediate Situation

- After some merging steps, we have some clusters



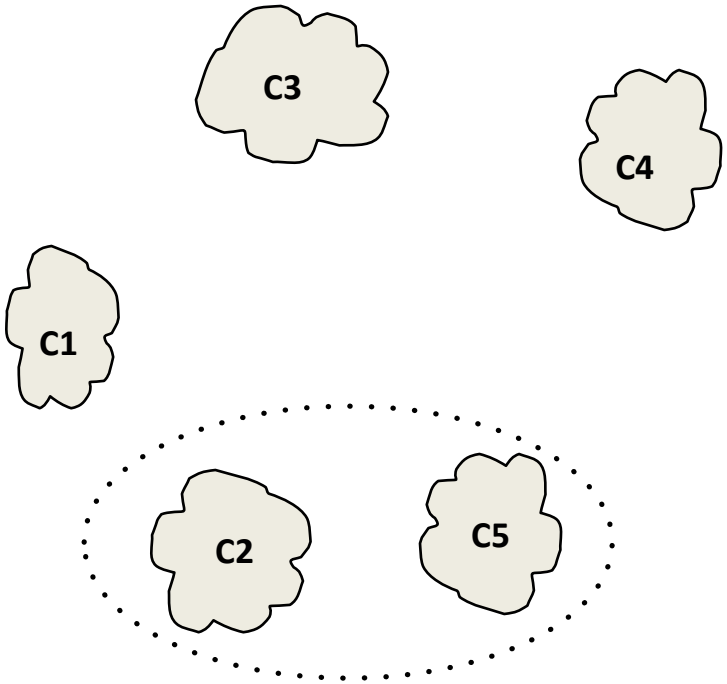

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

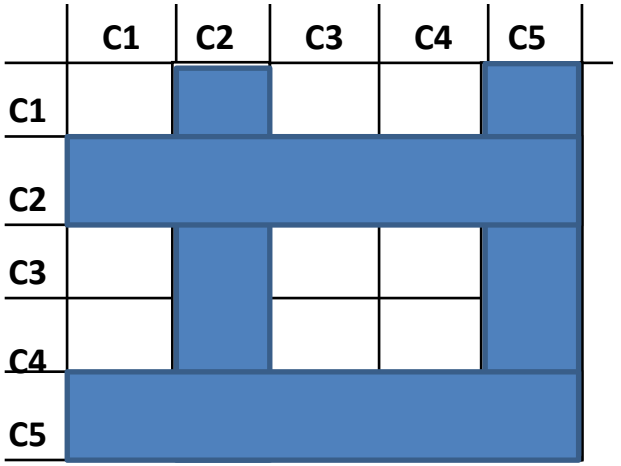
**Distance Matrix**



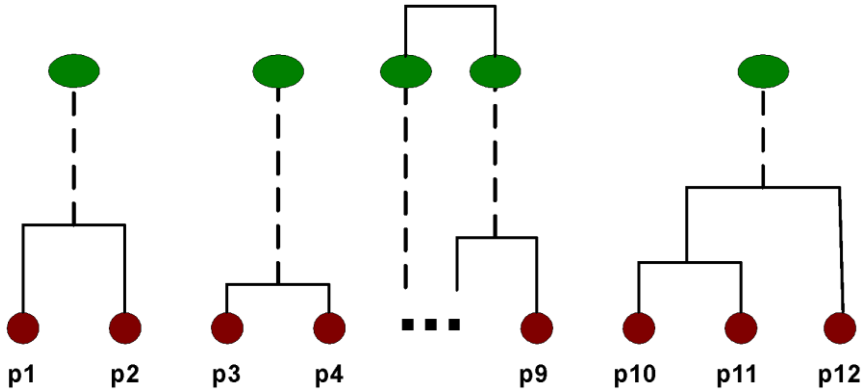
# Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the distance matrix.



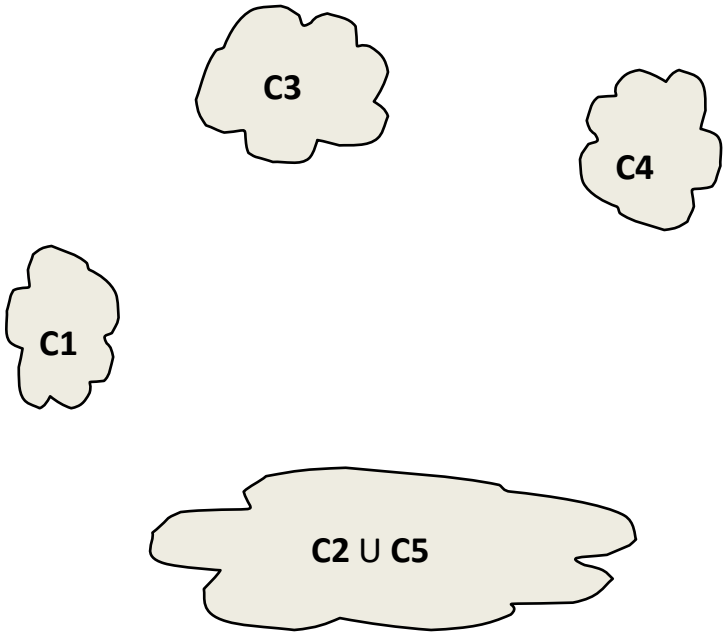




**Distance Matrix**



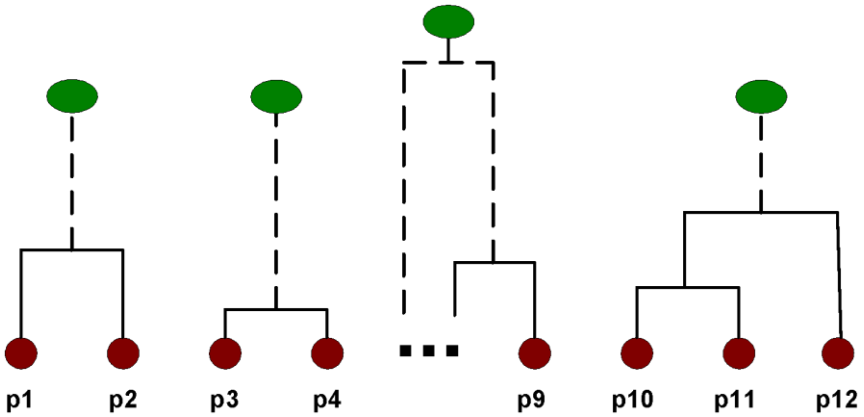
# After Merging

- The question is "How do we update the distance matrix?"

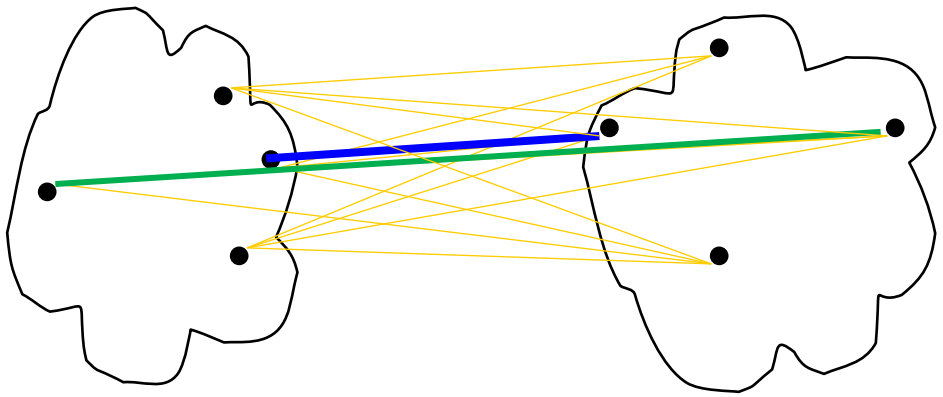



| | C1 | C2 U C5 | C3 | C4 |
|---|---|---|---|---|
| C1 | | ? | | |
| C2 U C5 | ? | | ? | ? |
| C3 | | ? | | |
| C4 | | ? | | |

**Distance Matrix**



# How to Define Inter-Cluster Distance



| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| | | | | | | |

**Distance Matrix**

- MIN
- MAX
- Group Average –UPGMA: **U**nweighted **P**air-**G**roup **M**ethod using an arithmetic **A**verage

# Group Average

- Distance between two clusters is the average of all pairwise distances between points in the two clusters.

$$\mathbf{distance}(\mathbf{Cluster}_i, \mathbf{Cluster}_j) = \frac{\sum_{\substack{p_i \in \mathbf{Cluster}_i \\ p_j \in \mathbf{Cluster}_j}} \mathbf{distance}(p_i, p_j)}{|\mathbf{Cluster}_i| * |\mathbf{Cluster}_j|}$$

# Example

| | Human | Chimpanzee | Gorilla | Orangutan | Gibbon |
|---|---|---|---|---|---|
| Human | | | | | |
| Chimpanzee | 1 | | | | |
| Gorilla | 4 | 2 | | | |
| Orangutan | 8 | 7 | 5 | | |
| Gibbon | 10 | 9 | 2 | 9 | |

The distances are real and determined based on the melting temperature of the DNA hybrids (the entire mitochondrial DNA)

# Distance matrix

Distance matrix

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | | | | | |
| B | 1 | | | | |
| C | 4 | 3 | | | |
| D | 8 | 7 | 2 | | |
| E | 10 | 9 | 4 | 6 | |

# Check: *Are the distances additive*?

Check each set of 4 points

ABCD

AC+BD=BC+AD=11>AB+CD=3

ABCE

AC+BE=AE+BC=13>AB+CE=5

ACDE

AD+CE=AE+CD=12>AC+DE=10

BCDE

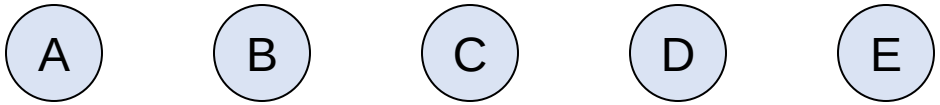
BD+CE=CD+BE=11>BC+DE=9

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | | | | | |
| B | 1 | | | | |
| C | 4 | 3 | | | |
| D | 8 | 7 | 2 | | |
| E | 10 | 9 | 4 | 6 | |

Yes, the distances are additive, we can build the tree

# UPGMA – demo 1/4

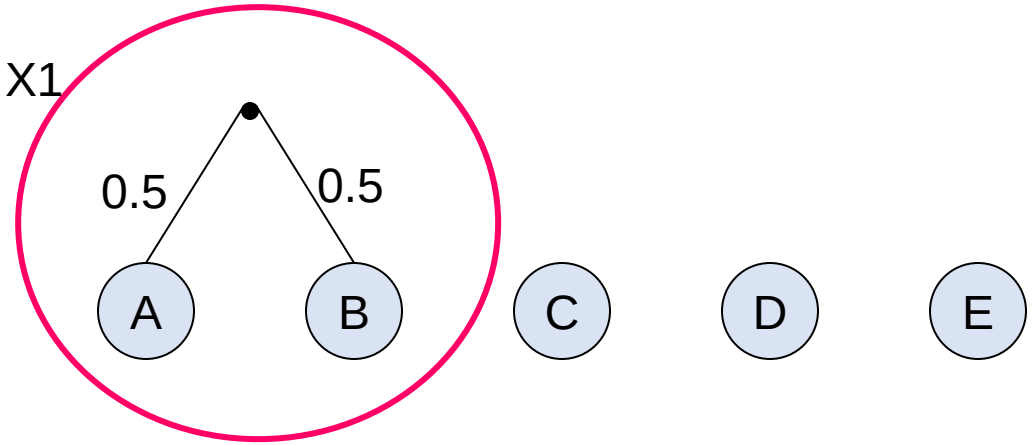
| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | | | | | |
| B | 1 | | | | |
| C | 4 | 3 | | | |
| D | 8 | 7 | 2 | | |
| E | 10 | 9 | 4 | 6 | |




Basic clusters – distance 0 between the elements inside each cluster

# UPGMA – demo 2/4

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |



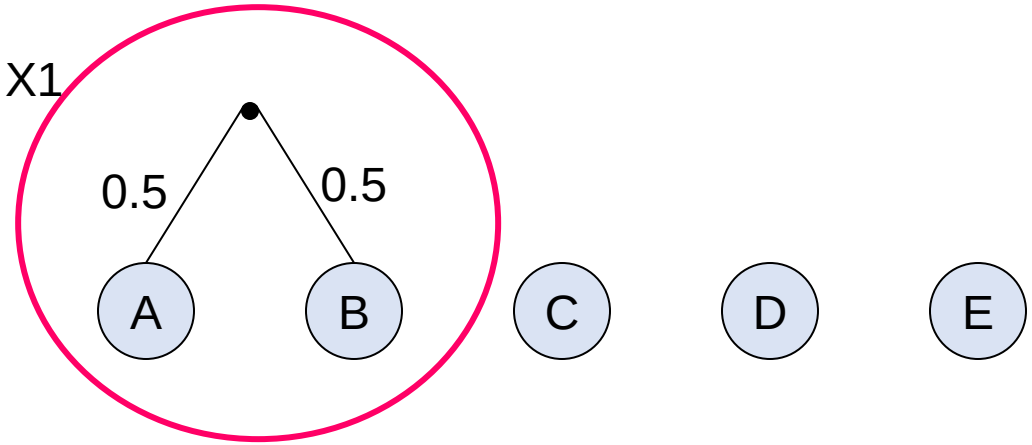

Form cluster X1 with min distance between two points

# UPGMA – demo 2/4

| | X1 | C | D | E |
|---|---|---|---|---|
| X1 | 0 | | | |
| C | 3.5 | 0 | | |
| D | 7.5 | 2 | 0 | |
| E | 9.5 | 4 | 6 | 0 |

Update distance matrix

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |




$dX_1C=(dAC+dBC)/(2)*1=3.5$

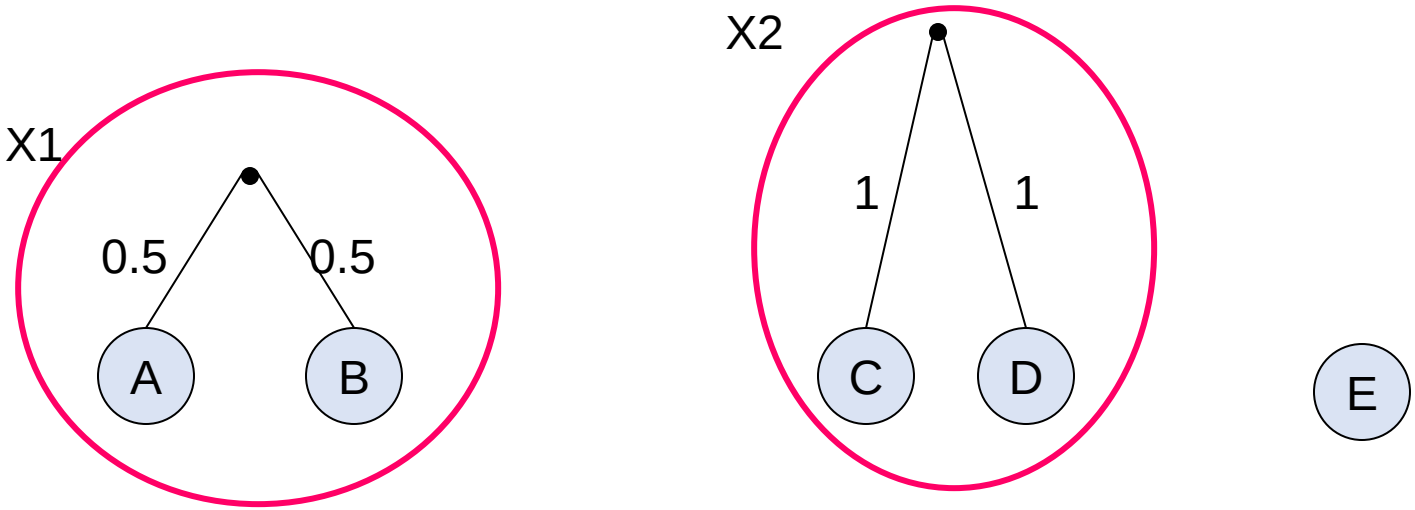
$dX_1D=(dAD+dBD)/(2)*1=7.5$

$dX_1E=(dAE+dBE)/(2)*1=9.5$

# UPGMA – demo 3/4

| | X1 | C | D | E |
|---|---|---|---|---|
| X1 | 0 | | | |
| C | 3.5 | 0 | | |
| D | 7.5 | 2 | 0 | |
| E | 9.5 | 4 | 6 | 0 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |




Form cluster X2 with min distance between two points

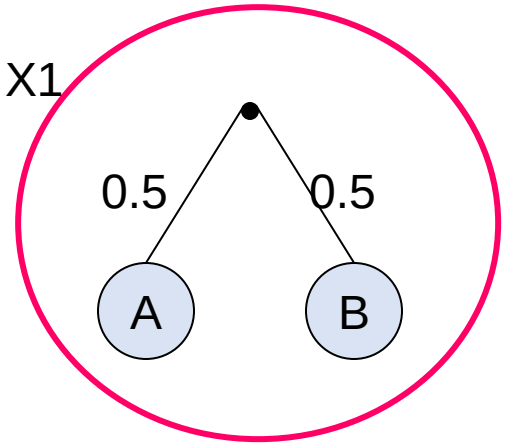
# UPGMA – demo 3/4

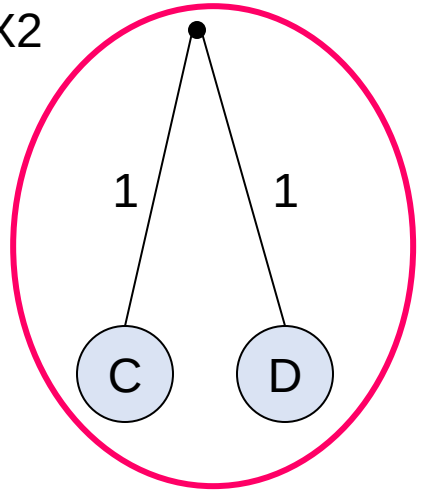
| | X1 | X2 | E |
|---|---|---|---|
| X1 | 0 | | |
| X2 | 5.5 | 0 | |
| E | 7.5 | 5 | 0 |

Update distance matrix

| | X1 | C | D | E |
|---|---|---|---|---|
| X1 | 0 | | | |
| C | 3.5 | 0 | | |
| D | 7.5 | 2 | 0 | |
| E | 9.5 | 4 | 6 | 0 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |





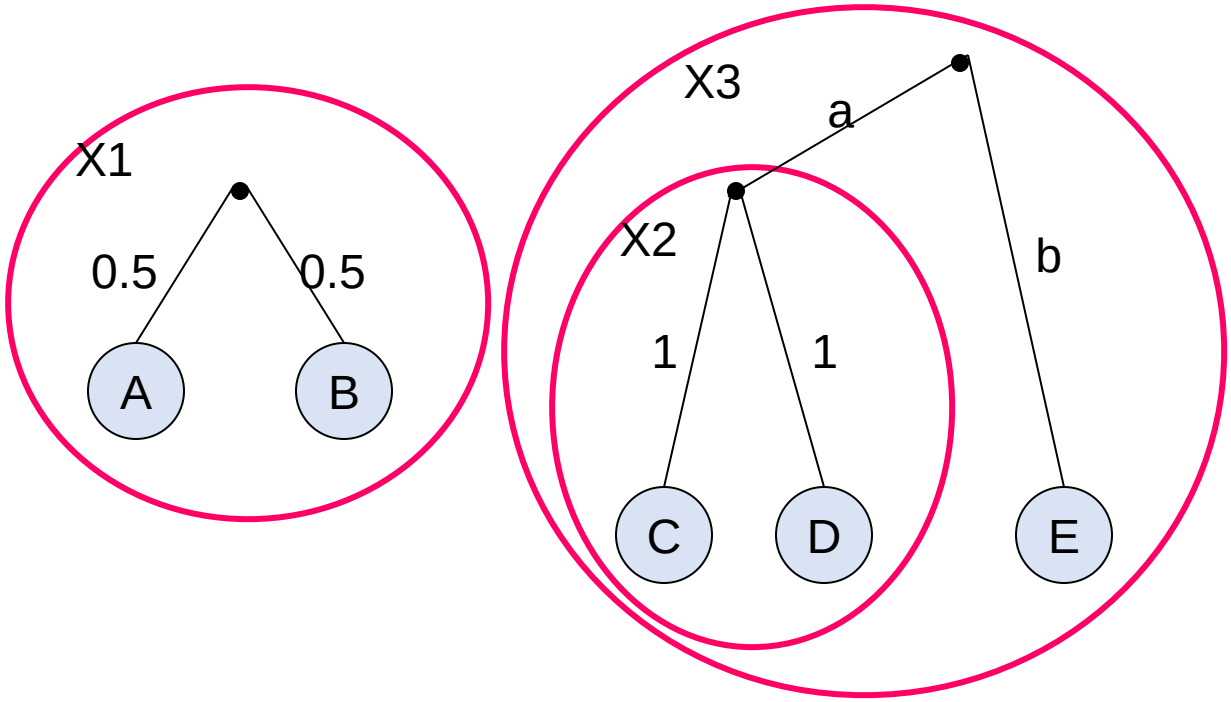
$dX_2E=(dCE+dDE)/(2)*1=5$

$dX_1X_2=(dAC+dBC+dAD+dBD)/(2*2)=5.5$

# UPGMA – demo 4/4

| | X1 | X2 | E |
|---|---|---|---|
| X1 | 0 | | |
| X2 | 5.5 | 0 | |
| E | 7.5 | 5 | 0 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |




a=1/2*5-1=1.5

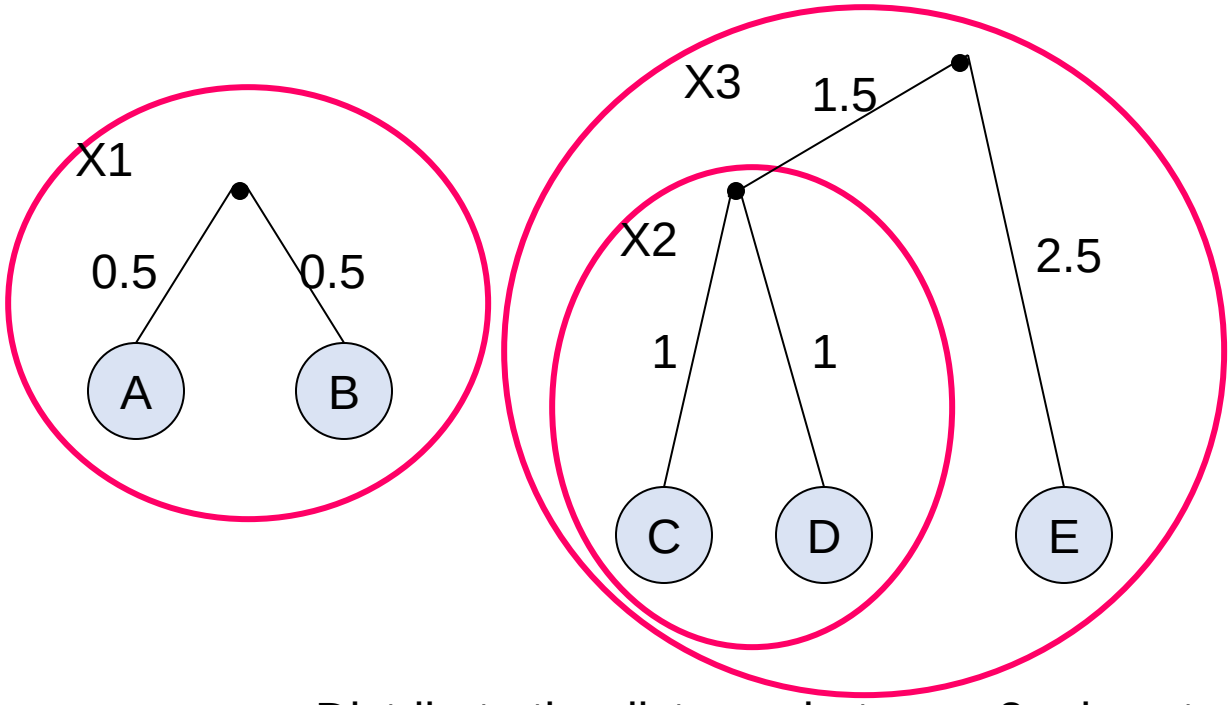
b=1/2*5=2.5

Create cluster X3 with min distance between 2 clusters

# UPGMA – demo 4/4

| | X1 | X2 | E |
|---|---|---|---|
| X1 | 0 | | |
| X2 | 5.5 | 0 | |
| E | 7.5 | 5 | 0 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |



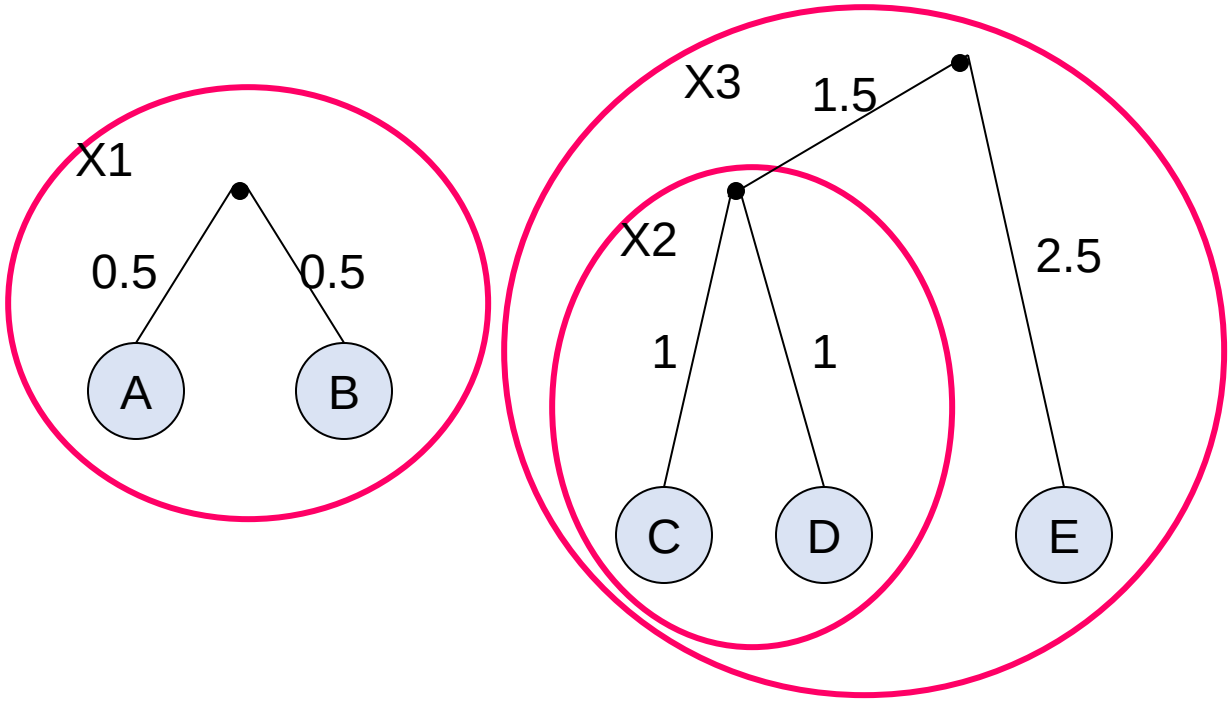

Distribute the distance between 2 edges to have half of this distance from the root to leaves in both branches

# UPGMA – demo 4/4

| | X1 | X3 |
|---|---|---|
| X1 | 0 | |
| X3 | 6.8 | 0 |

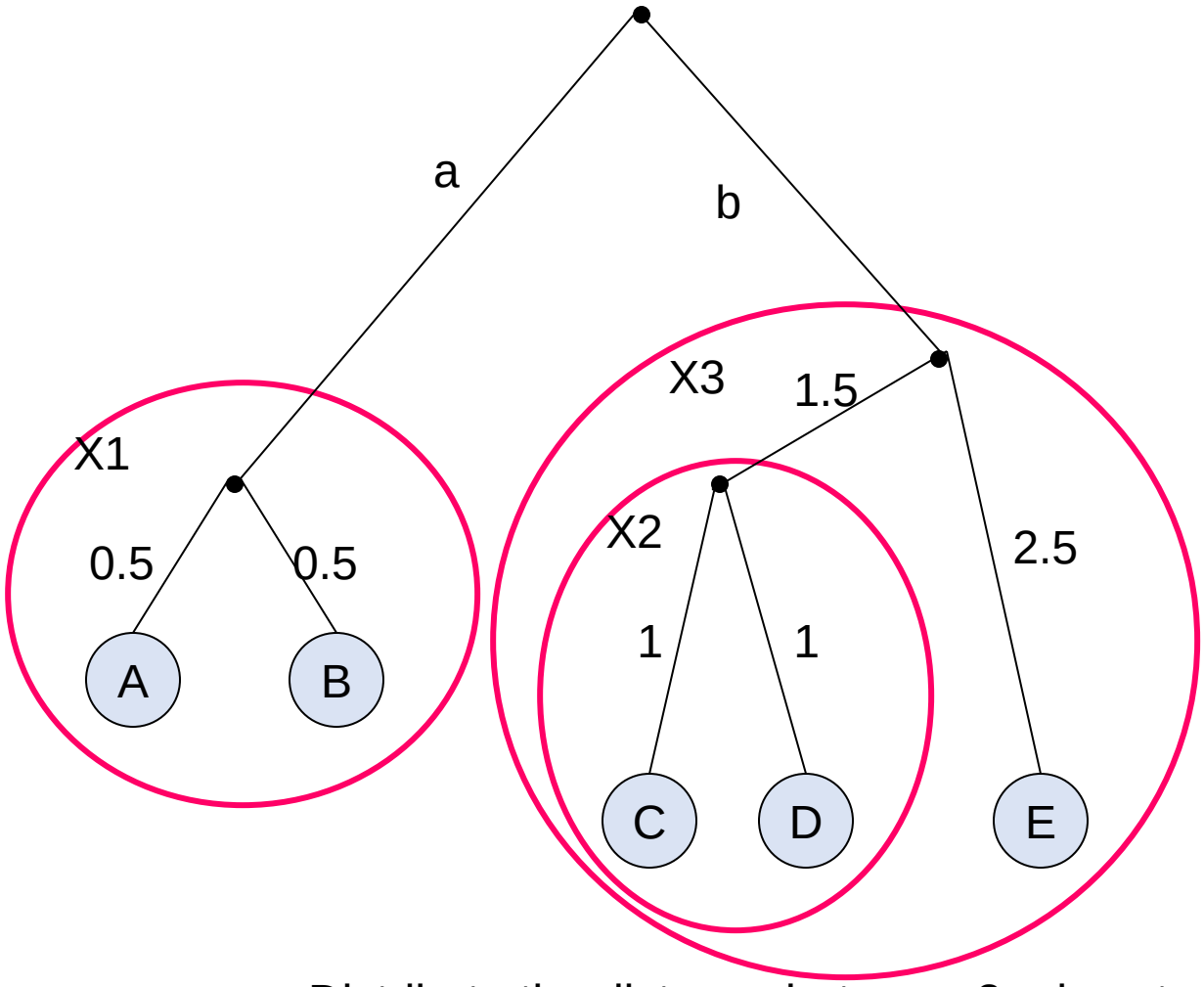
Update distance matrix

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |




dX1X3=(dAC+dAD+dAE+dBC+dBD+dBE)/2*3=(4+8+10+3+7+9)/6=6.8

# UPGMA – demo 4/4



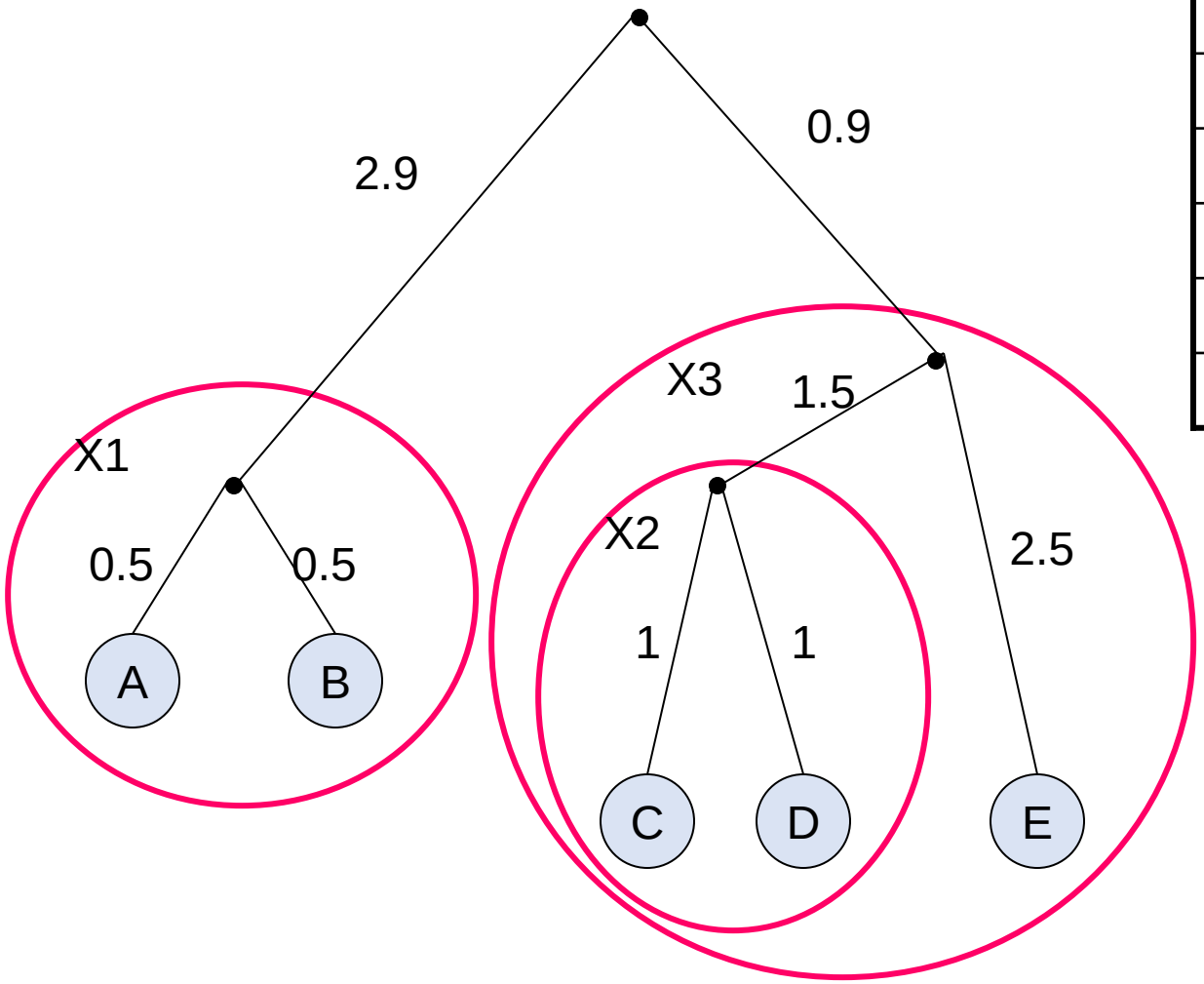

| | X1 | X3 |
|---|---|---|
| X1 | 0 | |
| X3 | 6.8 | 0 |

a=6.8/2-0.5=2.9

b=6.8/2-2.5=0.9

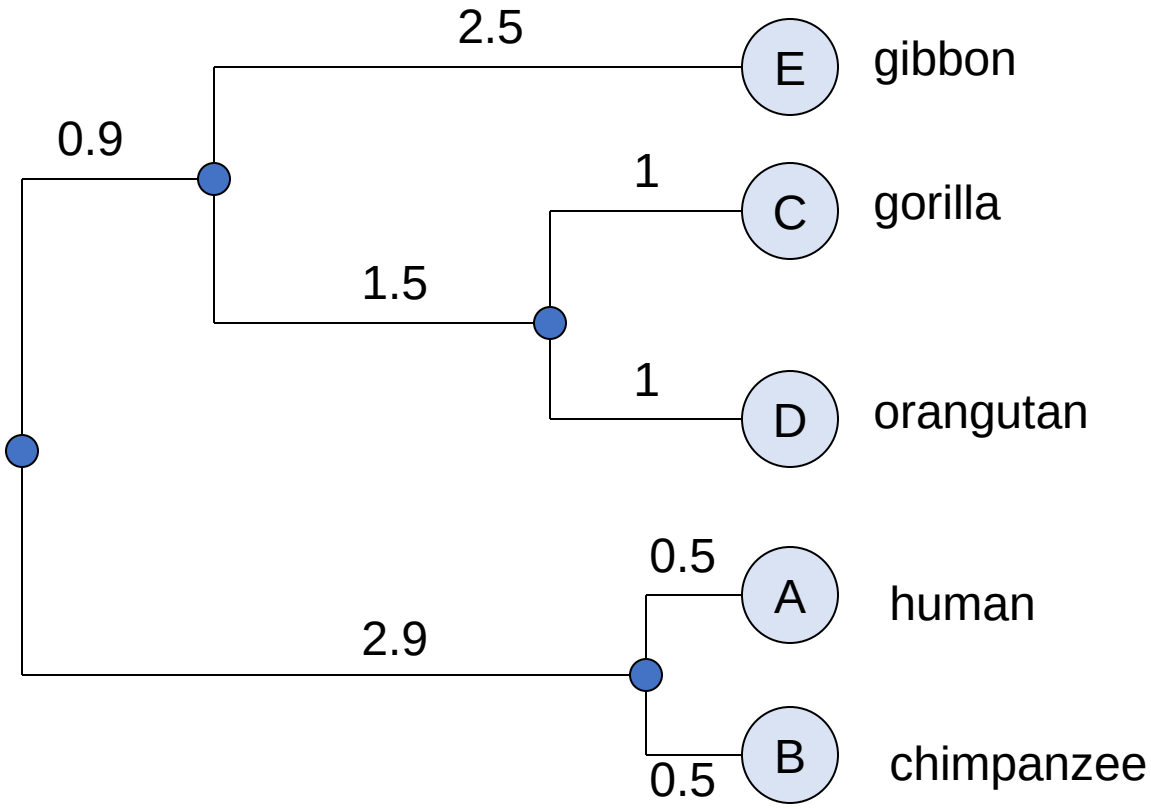
Distribute the distance between 2 edges to have half of this distance from the root to leaves in both branches

# UPGMA – the resulting tree

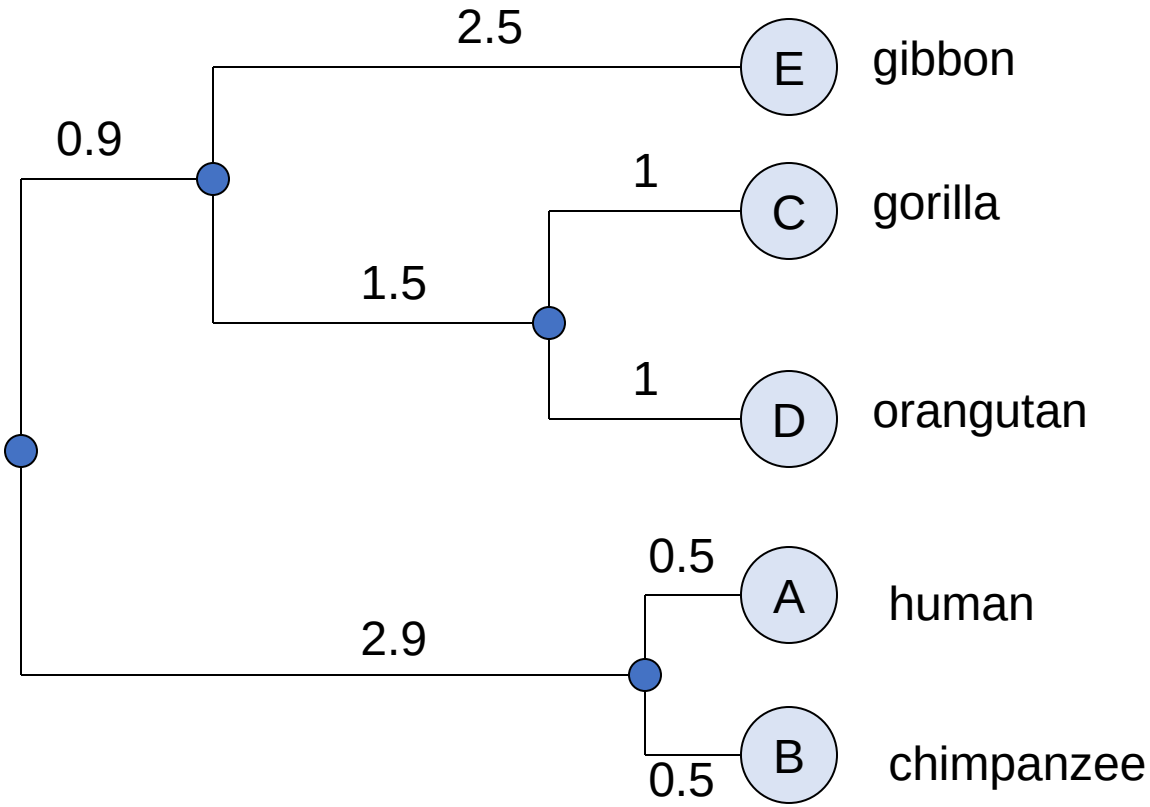



| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0 | | | | |
| B | 1 | 0 | | | |
| C | 4 | 3 | 0 | | |
| D | 8 | 7 | 2 | 0 | |
| E | 10 | 9 | 4 | 6 | 0 |

# The resulting tree with distances



# Molecular clock




The edge lengths can be viewed as times measured by *molecular clock* with a constant rate of mutational events.

We assume that divergence occurred at the same time at all branching points, the sum of edge lengths from any node to all leaves is the same for any possible path

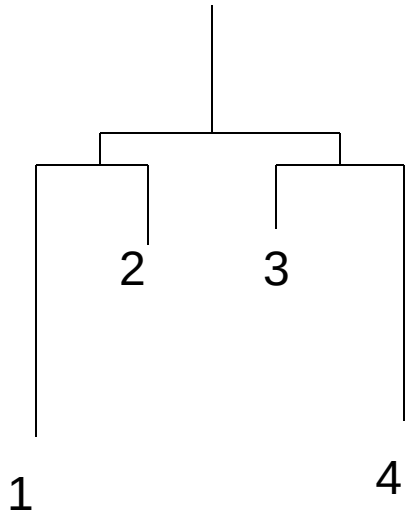
# Hierarchical Clustering: Time and Space

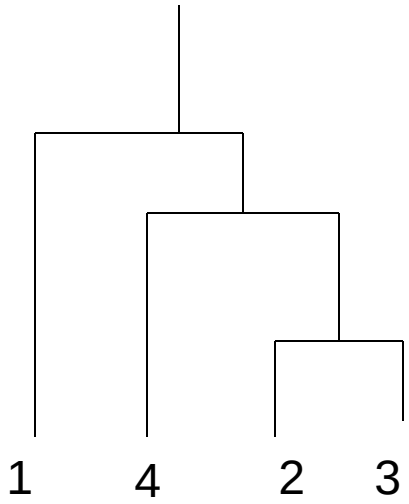
- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of data points.

- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, $N^2$, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time using more advanced data structures

**Hierarchical clustering algorithm is expensive !**

# When the tree does not reflect reality




- If the original tree, which we try to reconstruct, had different path lengths to its leaves, it will be reconstructed incorrectly by UPGMA.
- In this case, the closest leaves (2,3) are not siblings and they do not have a common parent, which will be assigned to them by UPGMA

# Predict correctness:
# Test for ultrametric condition

- We can predict whether the reconstruction of the real tree is likely to be correct by testing our distances for *ultrametric condition*:

  The distance matrix is *ultrametric* if for **any triplet** of sequences, Xi, Xj, Xk, the distances dij, dik, djk are either all equal or two are equal and the remaining one is smaller

Thus, if distances were derived from a real tree with a molecular clock, the distance matrix will be ultrametric

# Ultrametric and non-ultrametric distance matrices

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | | | |
| B | 1 | 0 | | |
| C | 4 | 2 | 0 | |
| D | 8 | 7 | 5 | 0 |

dAB=1, dAC=4, dBC=2

Non-ultrametric matrix ✖

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | | | |
| B | 4 | 0 | | |
| C | 2 | 4 | 0 | |
| D | 8 | 8 | 8 | 0 |

dAB=4, dAC=2, dBC=4 ✔

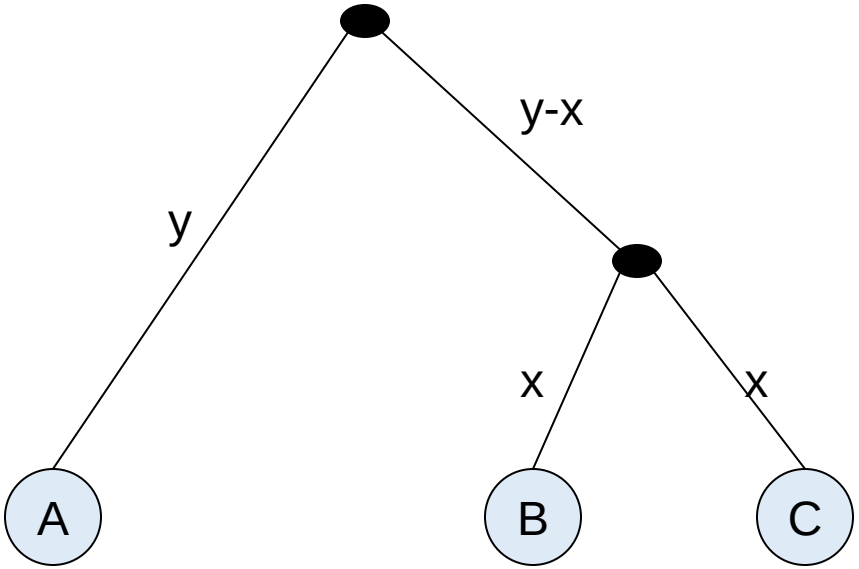
dAC=2, dAD=8, dCD=8 ✔

dBC=4, dBD=8, dCD=8 ✔

Ultrametric matrix

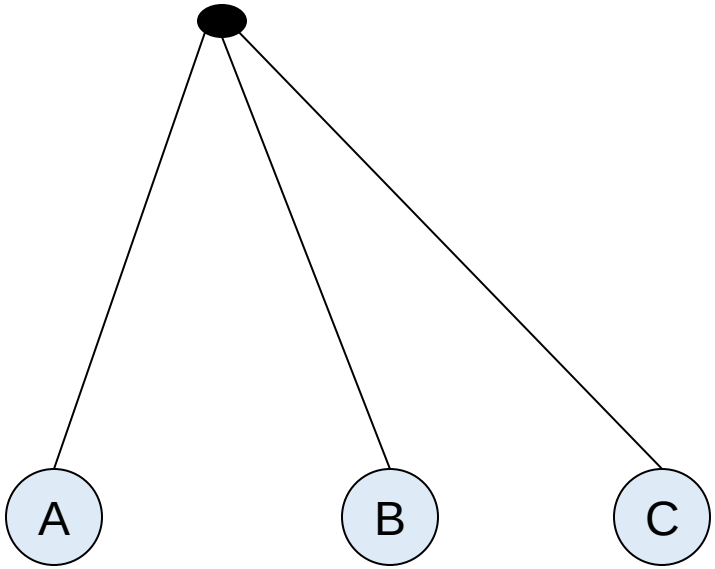
# Ultrametric trees: why

AB=AC>BC




or

AB=AC=BC




AB=y+y-x+x=2y

AC=2y

BC=2x, x<=y, since y-x>=0 (no negative edge lengths)

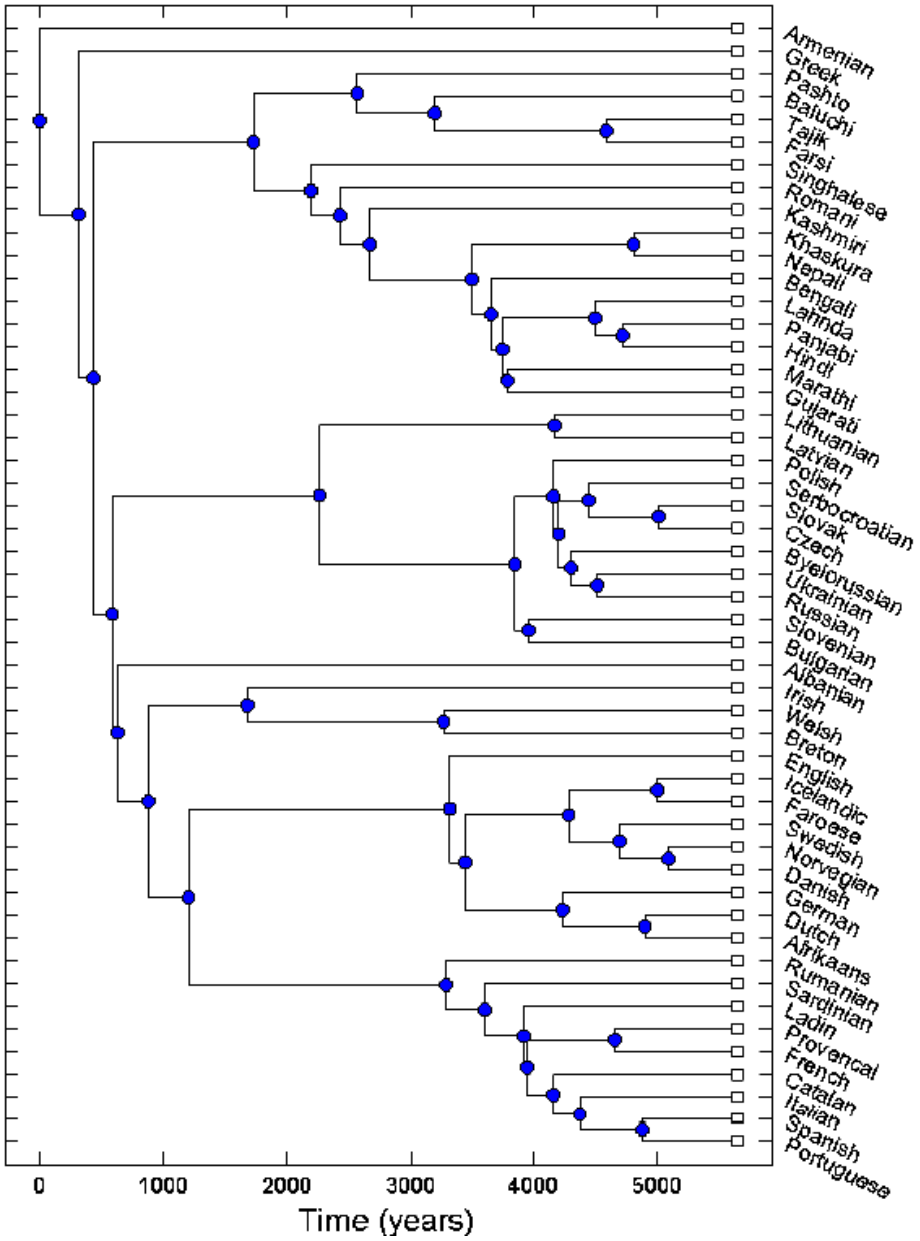
The rule for ultrametric trees:

2 out of 3 distances have a tie, and are >= than the third distance

# What to do if the distances are not ultrametric?

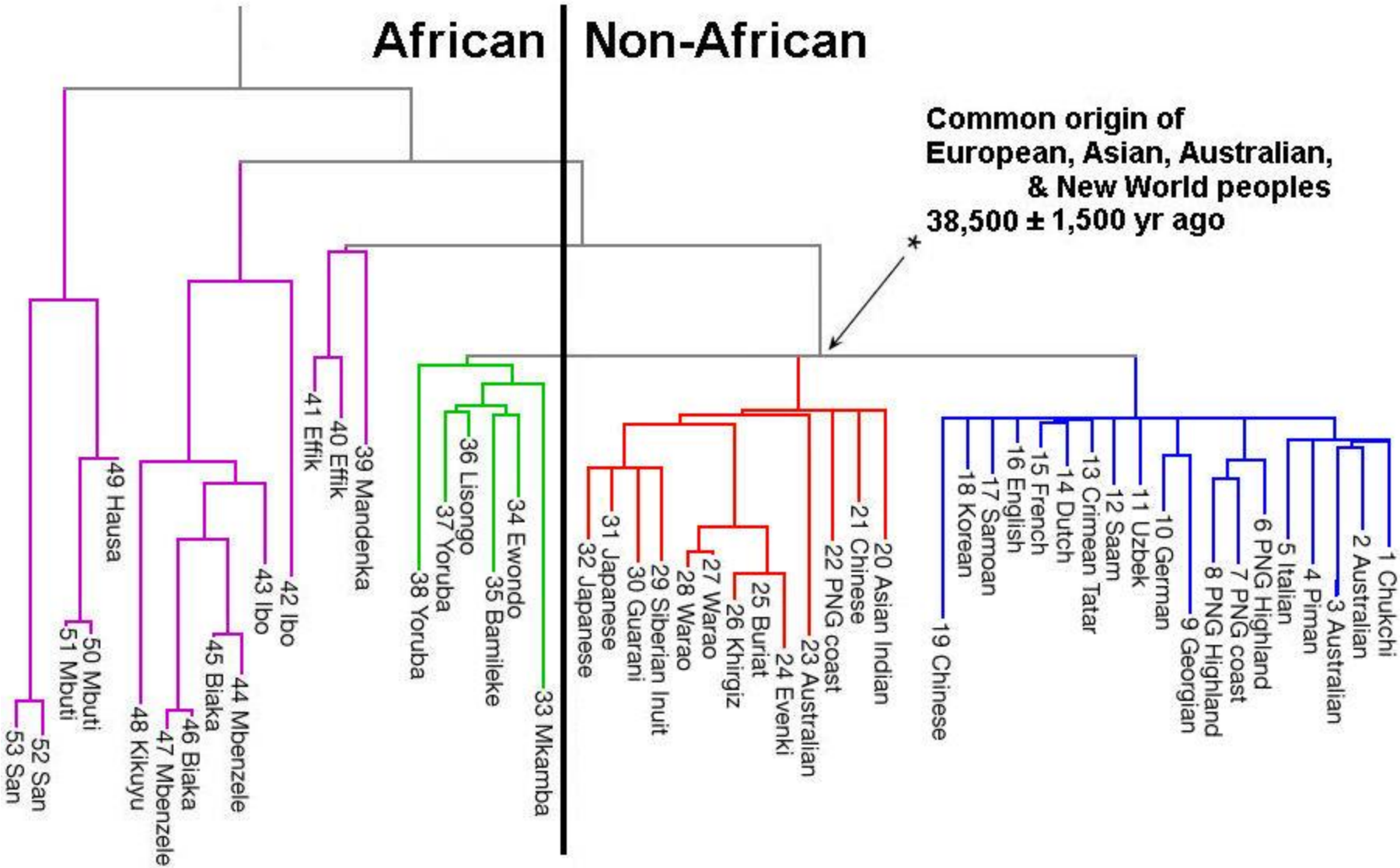
- We can apply UPGMA, but it may produce an incorrect tree.
- **The neighbor-joining algorithm** produces better results for non-ultrametric distances (see textbook)
- The main feature of the neighbor-joining algorithm is that we take into account not only how close are two clusters, but also how far away are they from other clusters
- This algorithm produces unrooted trees

# Evolution of languages

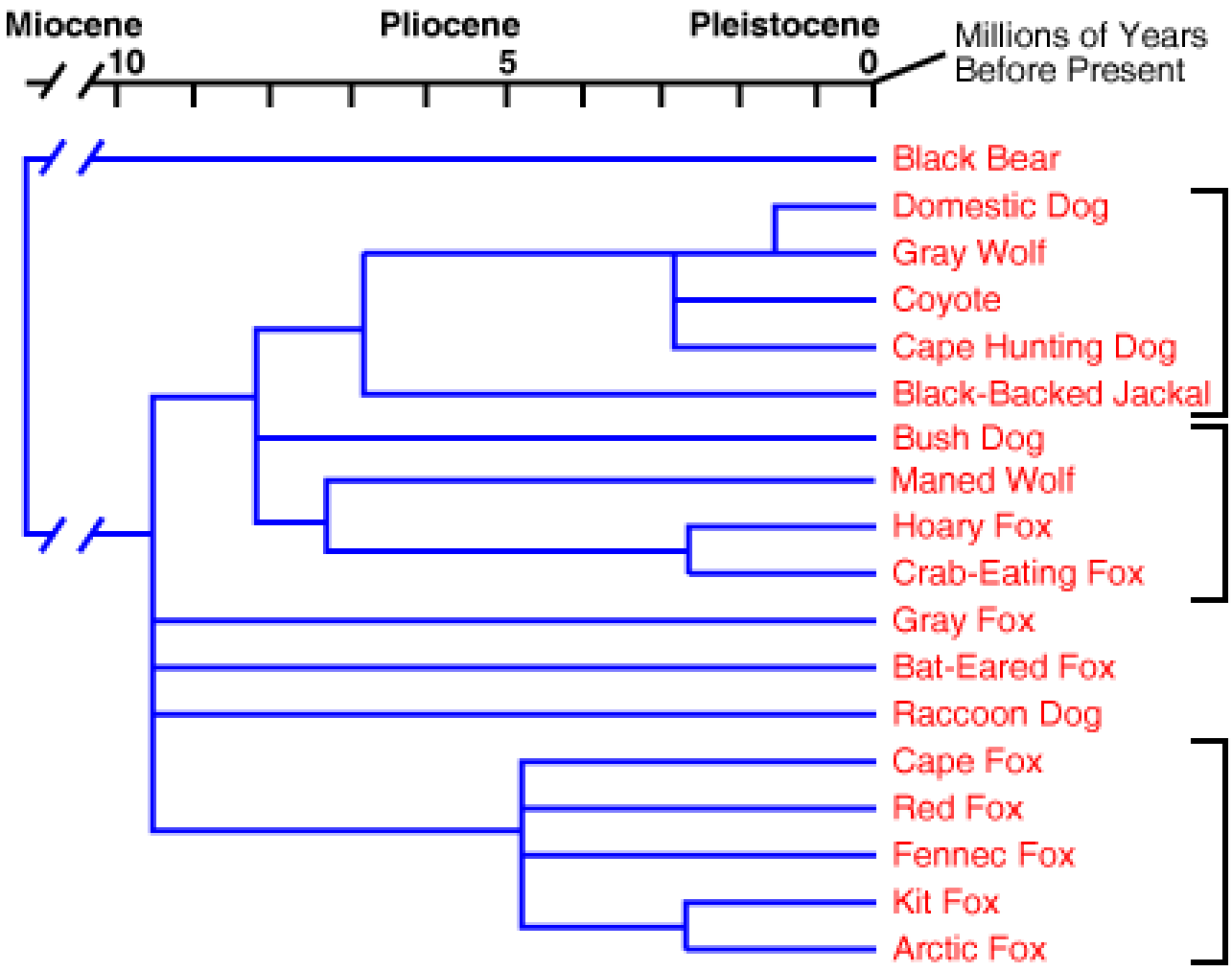



From
"Indo-European languages tree by Levenshtein distance" by M. Serva1 and F. Petroni

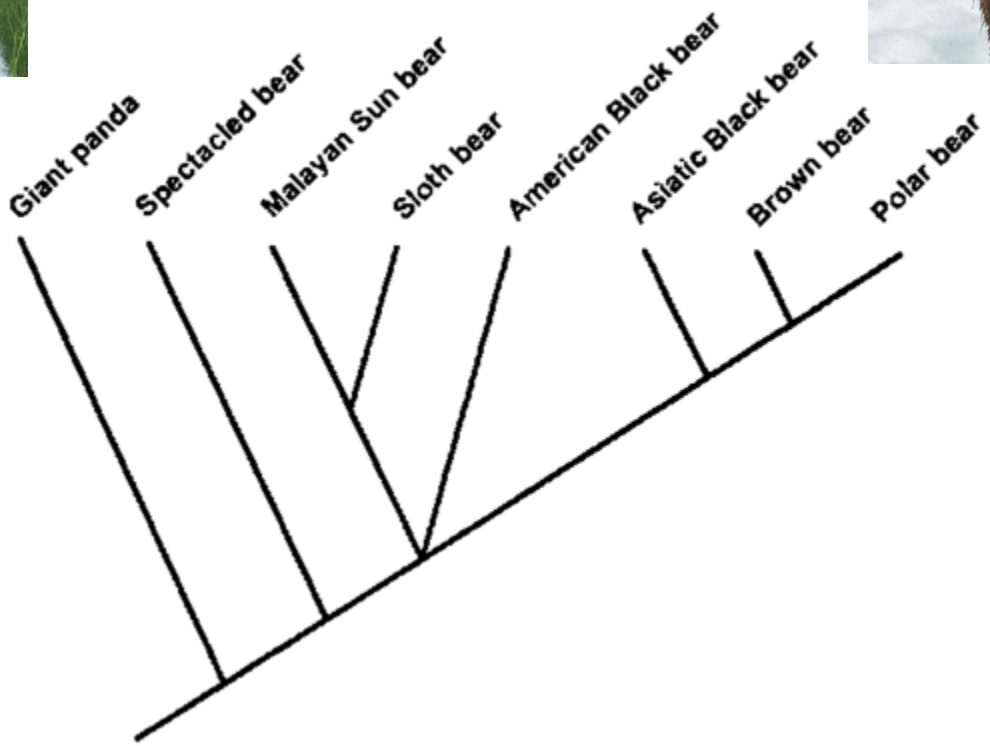
# Evolution of human populations



# Evolution of Canidae

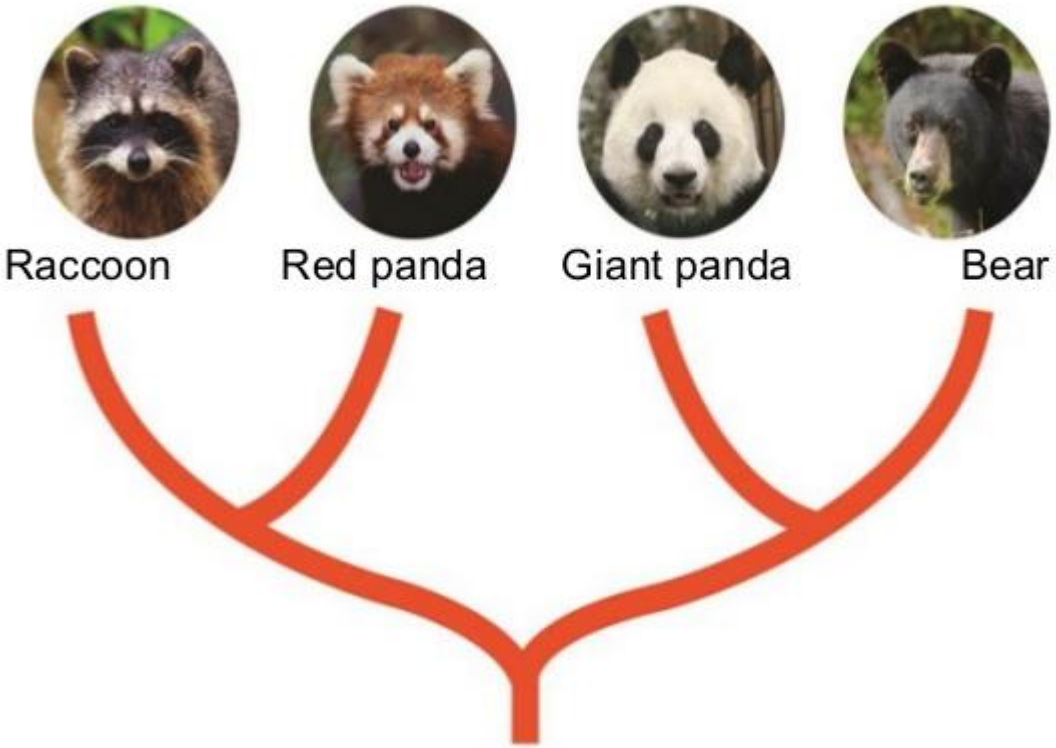



# Giant Panda is a bear



# What about Red Panda: a Cat or a Bear?

# Red Panda: a Bear or a Cat?



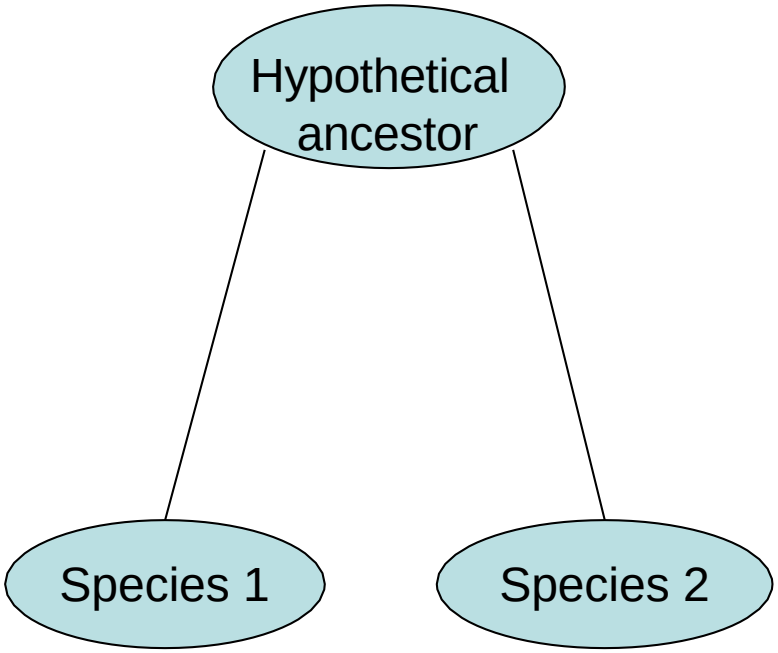

*Flynn, J. J.; Nedbal, M. A.; Dragoo, J. W.; Honeycutt, R. L. (2000). "Whence the Red Panda?" Molecular Phylogenetics and Evolution.*

# Phylogenetic trees

## Summary

# We are solving the following problem:

- Inferring phylogenies from a given data set



# Two main approaches to inferring phylogenies

- Distance-based: for example paiwise edit distance, score of pairwise alignment etc.
- Character-based: examine each character separately in a given site of a biological sequence

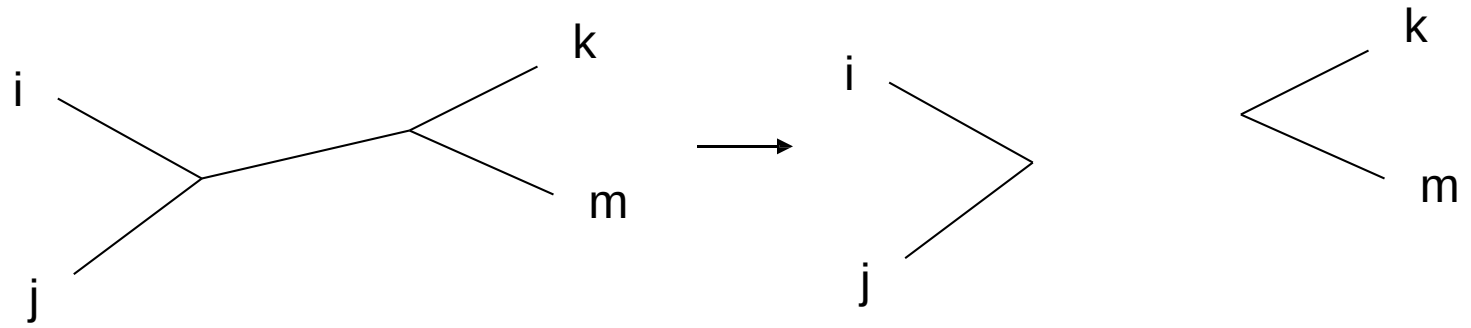
# Distance-based - recap

- Input: distance matrix of pairwise distances for N species
- Goal: find a tree consistent with the distance matrix. This means that the sum of edge lengths connecting each pair of leaves *ij* corresponds to a distance $M_{ij}$

# Additivity of distances

- For any 4 objects, i,j,k,m, there are 3 different pairwise sums:

  Dij+Dkm, Dik+Djm, Dim+Djk
- From these 3 sums, 2 should be equal and greater than the third – this will allow to group the smallest in between the 3 into a separate subtree

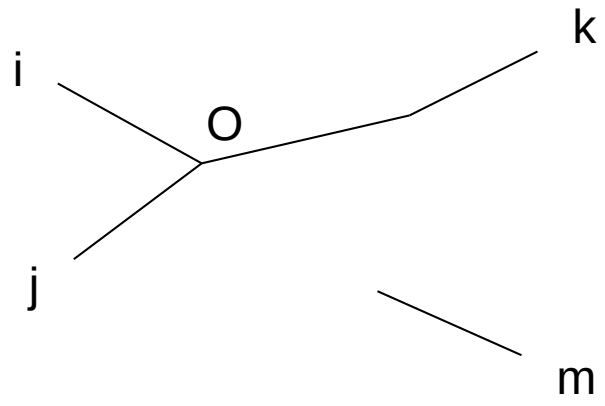



The distances are additive iff for any 4 objects there exits the following combination: Dij+Dkm<(Dim+Djk=Dik+Djm)

# Why the distances have to be additive

For 3 objects, any set of distances is OK:

Let iO=a, jO=b, kO=c, then Dij=a+b, Dik=a+c, Djk=b+c

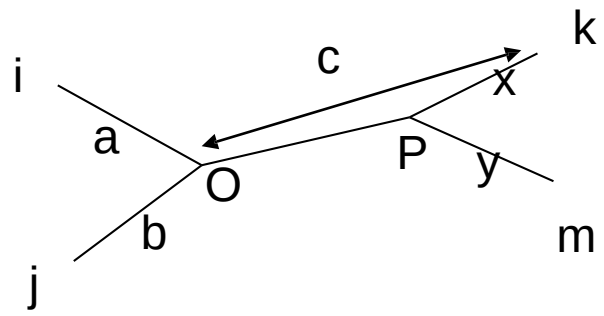


# Why the distances have to be additive

Let iO=a, jO=b, kO=c

Dij=a+b, Dik=a+c, Djk=b+c

Let's add the fourth object, m, to an arbitrary point P in the tree, and let mP=y, and kP=x, then OP=c-x




Then:

Dim+Djk=a+c-x+y+b+c=a+b+y+(2c-x)

Dik+Djm=a+c+b+c-x+y=a+b+y+(2c-x)

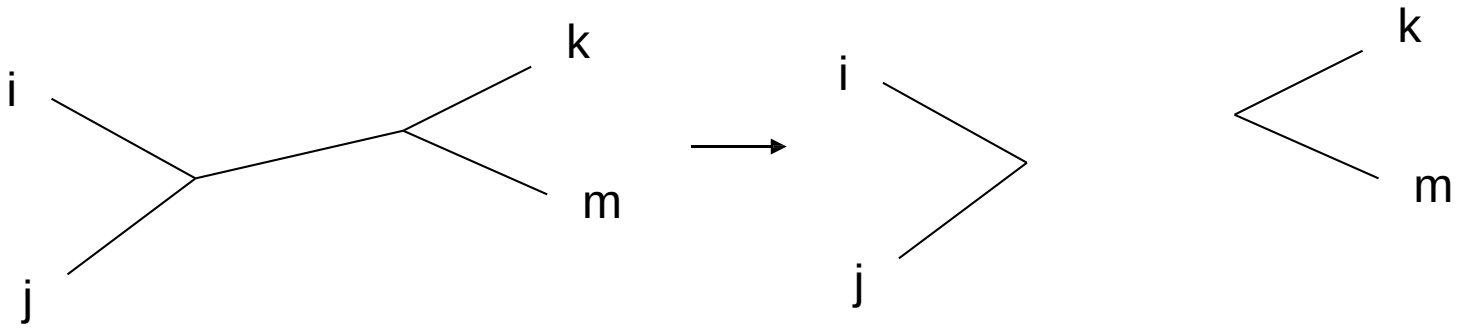
and

Dij+Dkm=a+b+x+y=a+b+y+x

2c-x>=x, since 2c>=2x

# The rule of additivity

- The distances are additive if, for any 4 objects, i,j,k,m,




Dij+Dkm<(Dim+Djk=Dik+Djm)

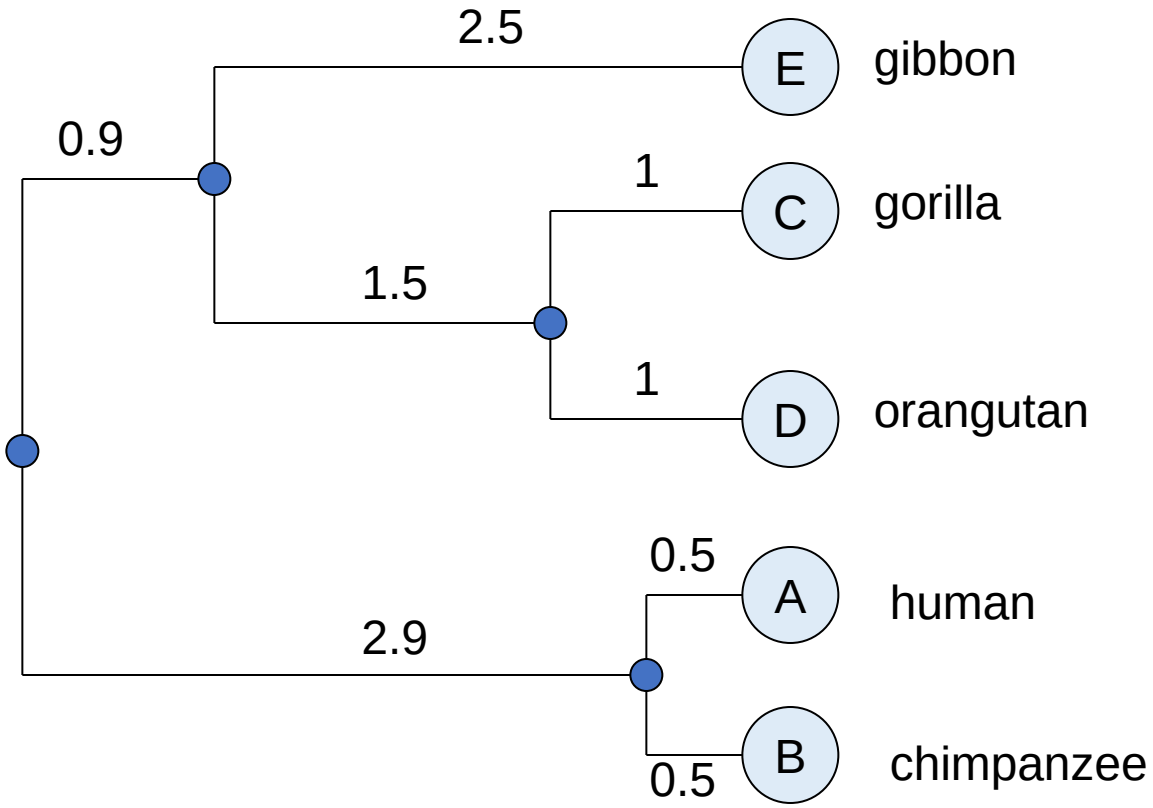
Without additive distances we CANNOT construct a phylogenetic tree

# UPGMA algorithm - summary

- Initialization:
  - Create N clusters, 1 species per cluster
  - Set the size of each cluster to 1
  - Create leaf for each cluster
- Iterate (until only 1 cluster left)
  - Find Ci and Cj with min dCiCj
  - Create a new cluster C(ij) which has n(ij)=ni+nj members
  - Connect Ci and Cj through a new parent node and set the distance from this new parent node to the leaf node of each cluster to ½ dCiCj
  - Delete columns and rows that correspond to amalgamed clusters i and j
  - Add a column and a row for a new cluster
  - Compute distances from a new cluster C(ij) to all remaining clusters:

$$dC(ij)Ck=\Sigma_{\text{all } x\in C(ij),\ \text{all } y\in Ck}\ dXY/(n(ij)*nk)$$

# Ultrametric trees and UPGMA




This tree is clocklike, ultrametric: the total length of the path from a given internal node to each leaf is the same.

Assumption: the molecular clock of mutations ticks with a constant pace

- UPGMA reconstructs the tree based on the molecular clock assumption, that is why a new node is always created at the same distance from all the leaves

# When the tree reflects reality

$$SSQ(T)=\Sigma_{i \text{ from } 1 \text{ to } N}\Sigma_{j\neq i}w_{ij}(d_{ij}-TreeD_{ij})$$

- If the solution to SSQ(T)=0, and there was a molecular clock with constant pace, then UPGMA guarantees to find the correct solution

- If not:
  - It can find a good enough solution, but the correctness of the tree topology is not guaranteed
  - Use *the neighbor-joining* algorithm to check the correctness of the tree topology. This algorithm relies on additivity but does not require the distances to be ultrametric

# Test for ultrametric condition

- We can predict whether the reconstruction of the real tree is likely to be correct by testing our distances for *ultrametric condition*:

  The distance matrix is *ultrametric* if for any triplet of sequences, Xi, Xj, Xk, the distances dij, dik, djk are either all equal or two are equal and the remaining one is smaller

Thus, if distances were derived from a real tree with a molecular clock, the distance matrix will be ultrametric