# Building Phylogenetic Trees 2

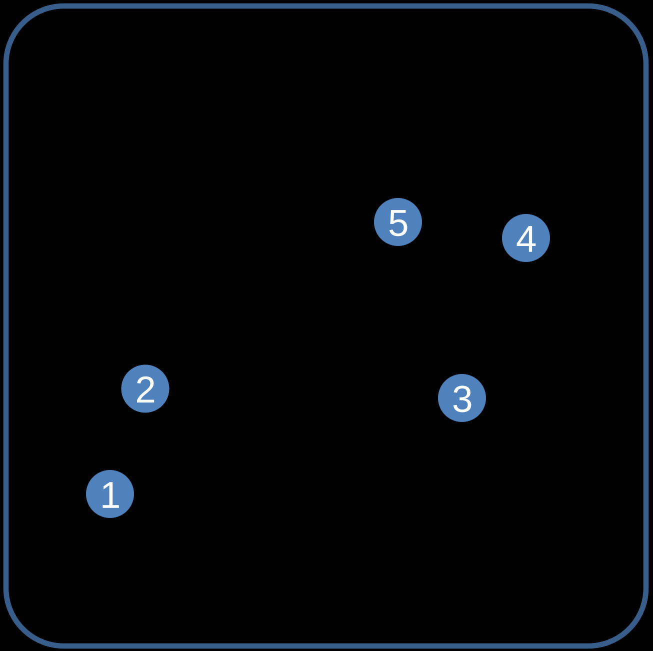
## Change in Implementation Project

- Before: “Identify a phylogenetic tree with parsimony superior to that of a UPGMA tree.”
- Now: “Compute a UPGMA tree and a Neighbor joining tree for any multiple sequence alignment”

- Why?
  - Slippery grading slope
  - If you work really hard to do a good job finding a parsimonious tree (and you can work really hard on this), depending on testing input, I might never notice that you did – and not give you credit

## The UPGMA process

Input: A distance matrix



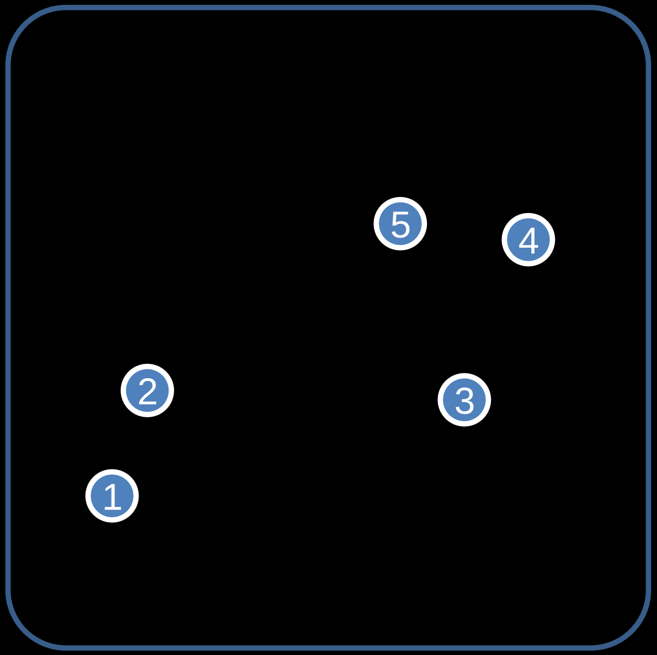
Sequences represented by points in space.

Distances between points represent sequence similarity, specified with the distance matrix

# The UPGMA process

Input: A distance matrix

Step 1) make each sequence a member of its own group



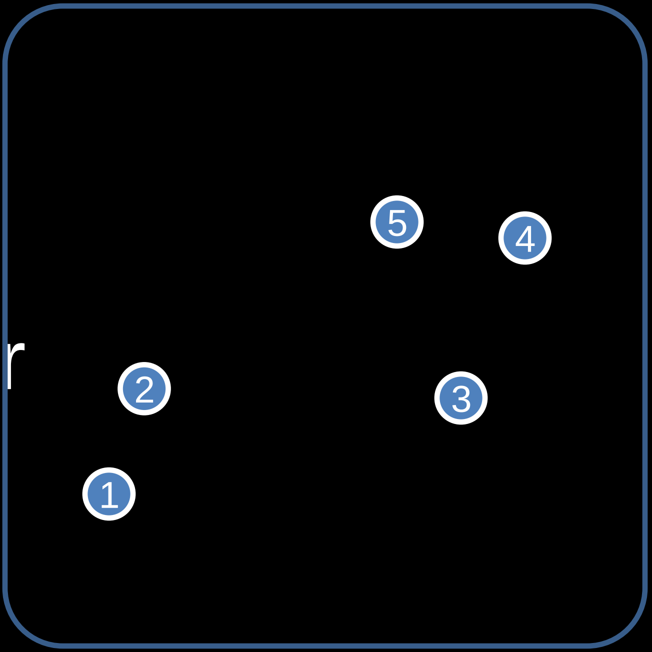
# The UPGMA process

Input: A distance matrix

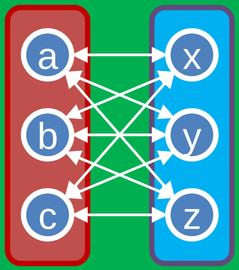
Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the closest pair of clusters



The distance between two clusters is the average of the pairwise distances



Distance((abc), (xyz)) is the average of d(a,x) d(a,y), d(a,z), d(b,x), d(b,y), d(b,z), d(c,x), d(c,y), d(c,z)

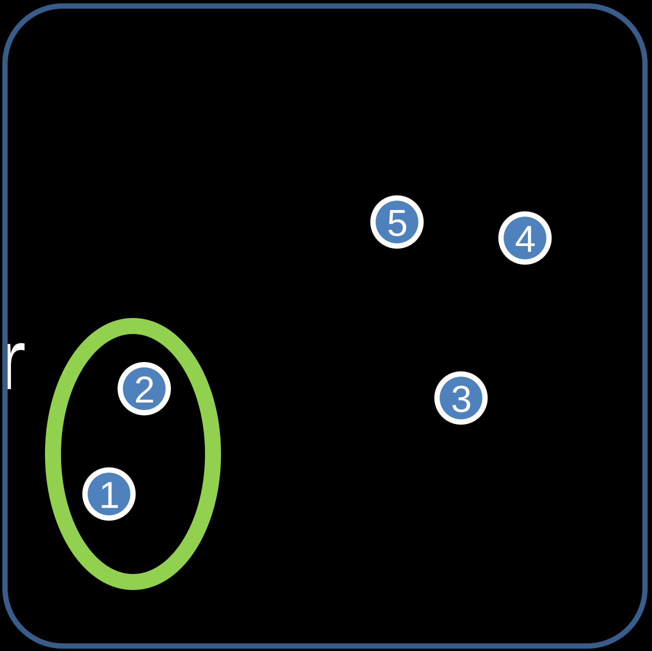
## The UPGMA process

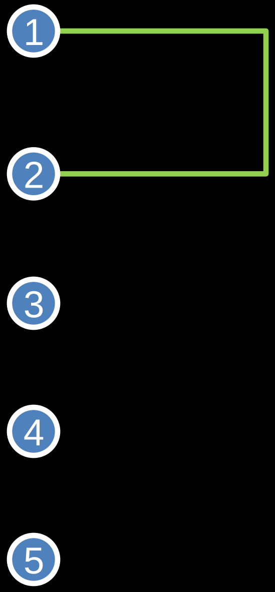
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the closest pair of among the active clusters (1,2)
- Step 3) Group the closest pair in a new cluster





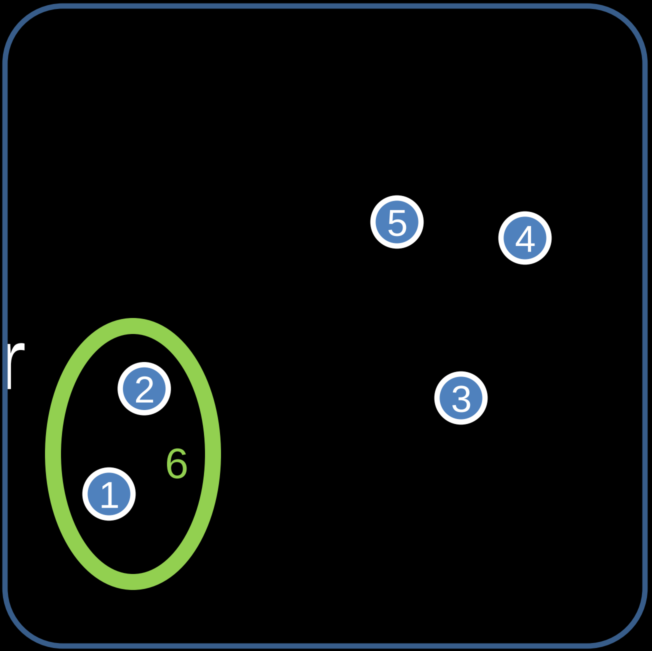
# The UPGMA process

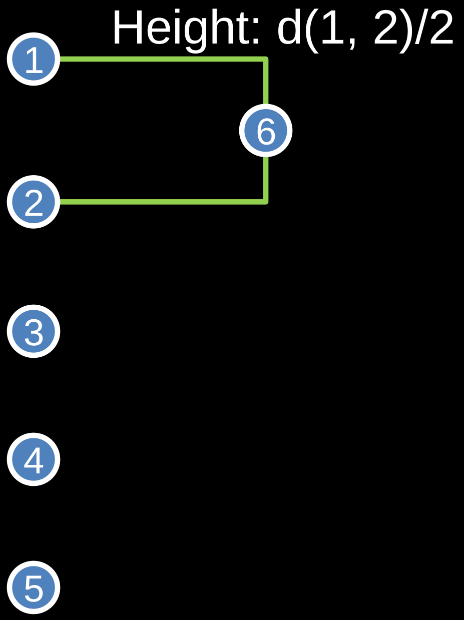
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the closest pair of among the active clusters (1,2)
- Step 3) Group the closest pair in a new cluster
- Step 4) Set height of hypothetical shared ancestor at d(1,2)/2





# The UPGMA process

Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

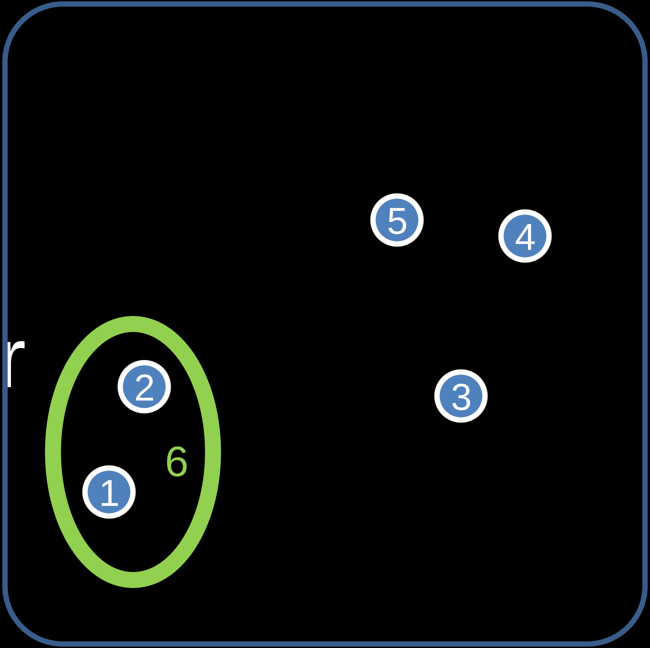
While (num active clusters > 1)

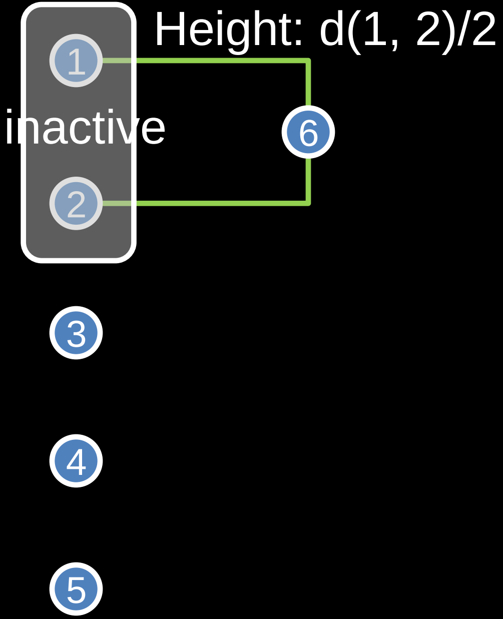
  Step 2) Find the closest pair of among the active clusters (1,2)

  Step 3) Group the closest pair in a new cluster

  Step 4) Set height of hypothetical shared ancestor at d(1,2)/2

  Step 5) Remove the old clusters 1,2, add the new one (6) to list of active clusters





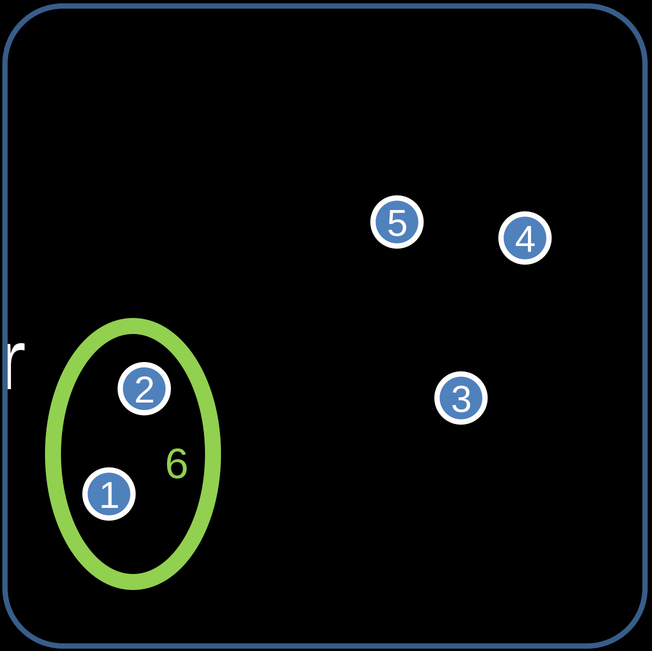
# The UPGMA process

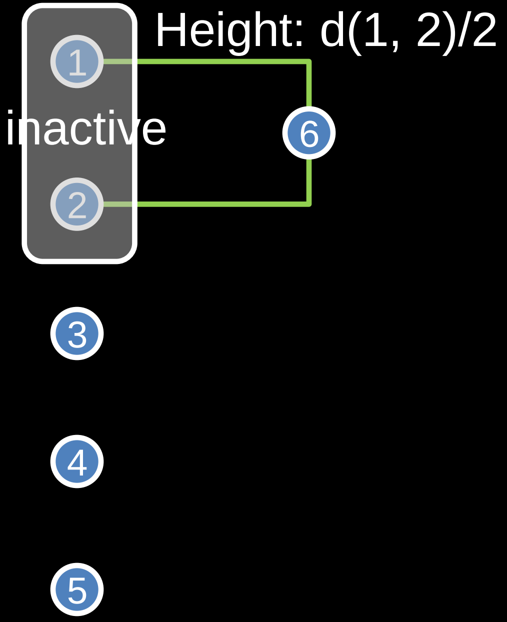
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

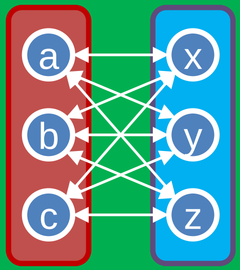
While (num active clusters > 1)

Step 2) Find the new closest pair among the active clusters (4,5)





The distance between two clusters is the average of the pairwise distances



Distance((abc), (xyz)) is the average of d(a,x) d(a,y), d(a,z), d(b,x), d(b,y), d(b,z), d(c,x), d(c,y), d(c,z)

# The UPGMA process

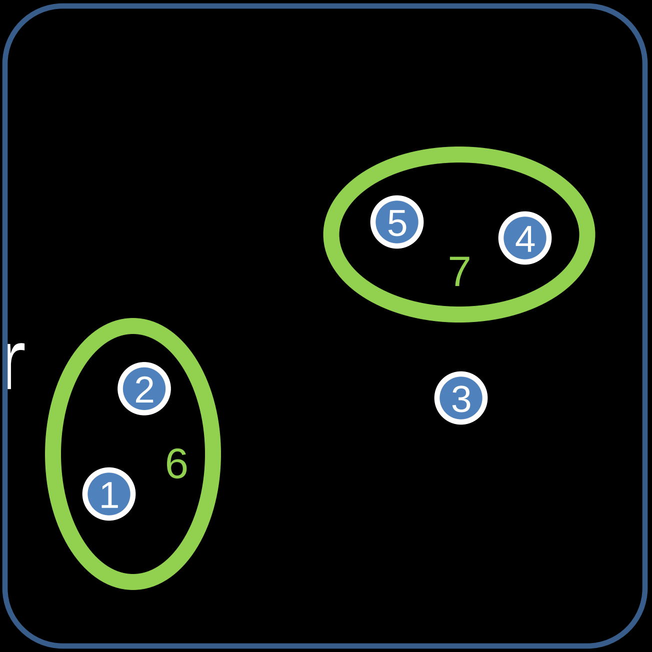
Input: A distance matrix

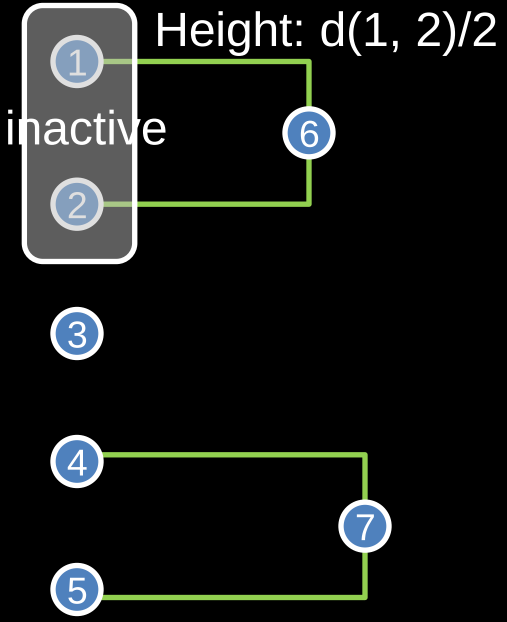
Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

  Step 2) Find the new closest pair among the active clusters (4,5)

  Step 3) Group the closest pair in a new cluster





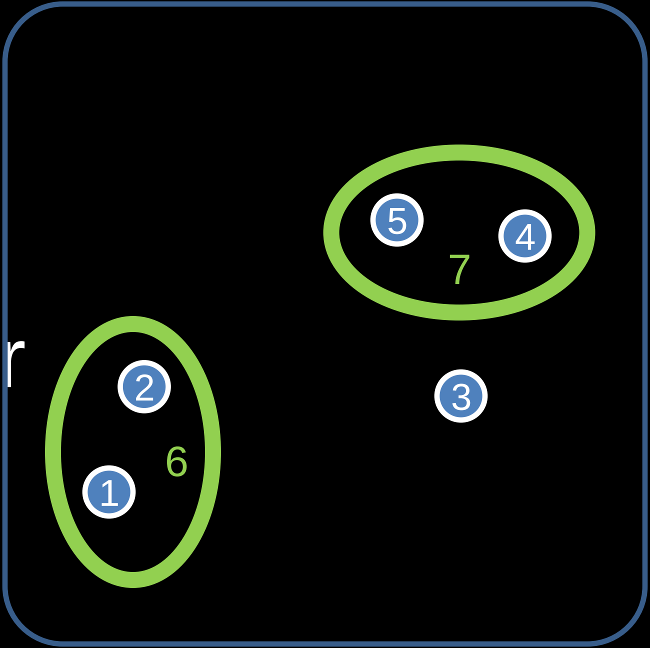
# The UPGMA process

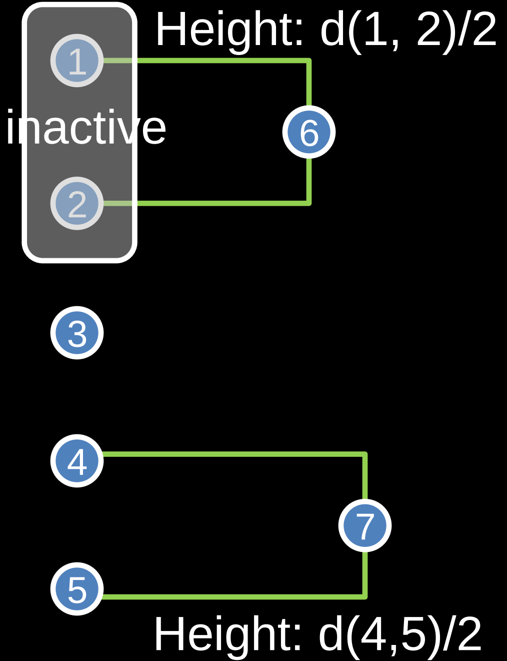
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (4,5)
- Step 3) Group the closest pair in a new cluster
- Step 4) Set height of hypothetical shared ancestor at d(4,5)/2





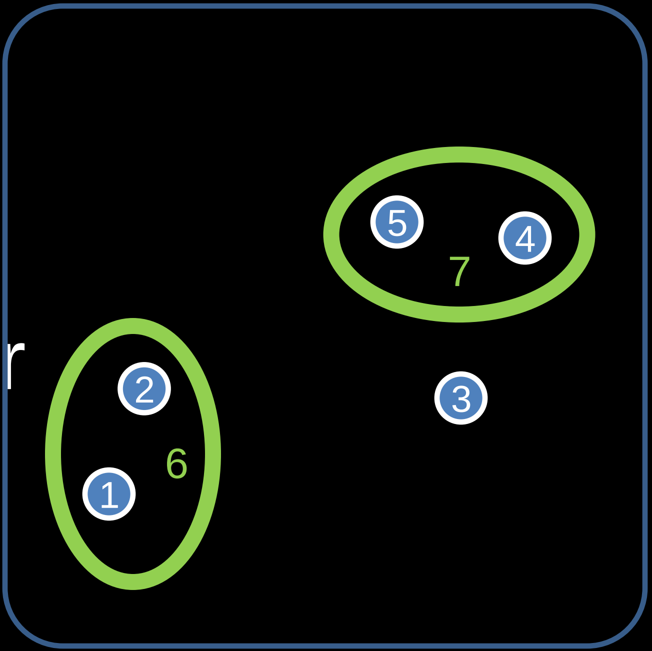
# The UPGMA process

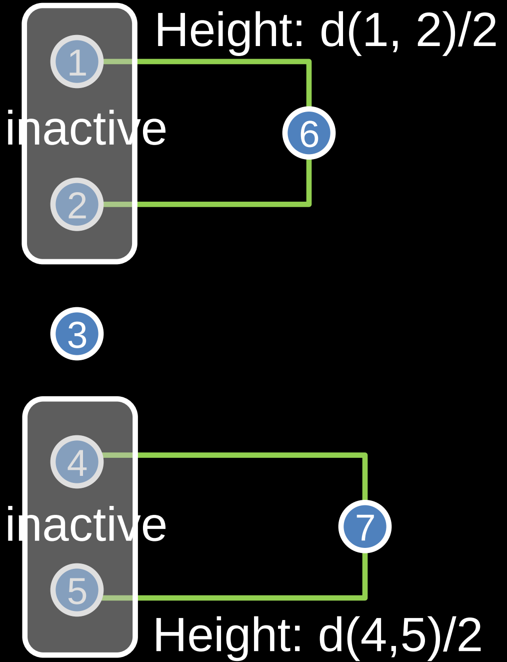
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (4,5)
- Step 3) Group the closest pair in a new cluster
- Step 4) Set height of hypothetical shared ancestor at d(4,5)/2
- Step 5) Remove the old clusters 4,5, add the new one (6) to list of active clusters





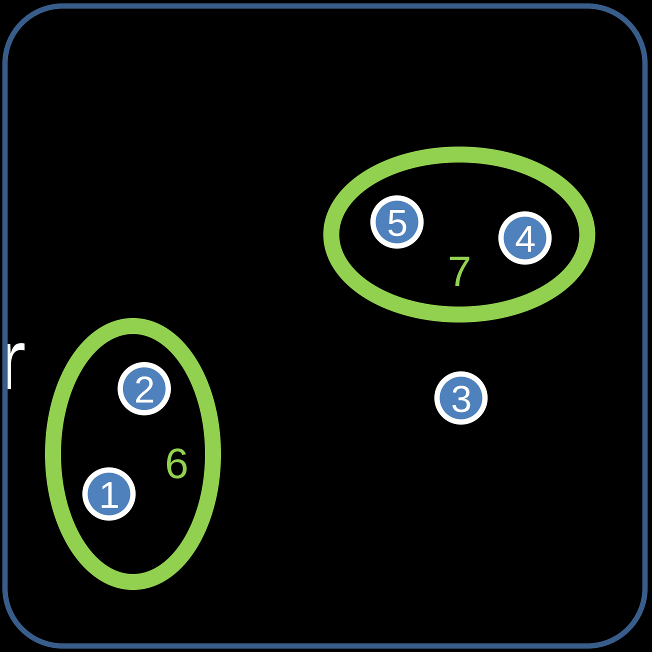
# The UPGMA process

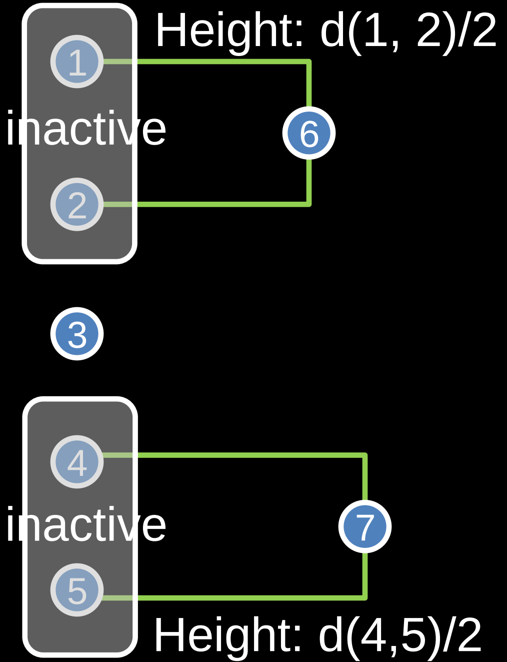
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

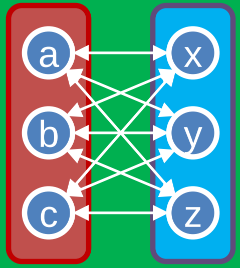
While (num active clusters > 1)

Step 2) Find the new closest pair among the active clusters (3,7)





The distance between two clusters is the average of the pairwise distances



Distance((abc), (xyz)) is the average of d(a,x) d(a,y), d(a,z), d(b,x), d(b,y), d(b,z), d(c,x), d(c,y), d(c,z)

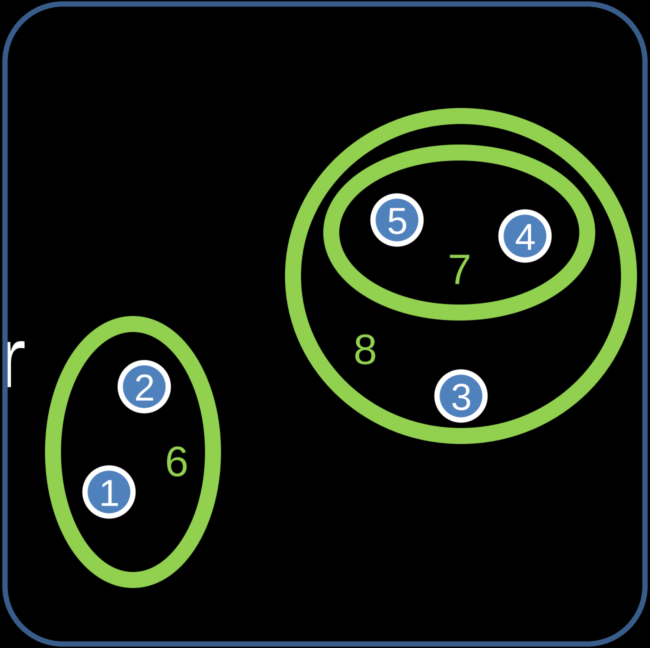
# The UPGMA process

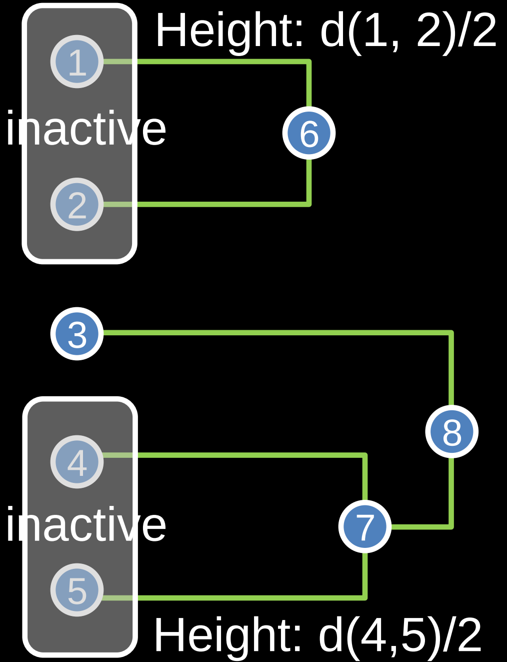
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (3,7)
- Step 3) Group the closest pair in a new cluster (8)





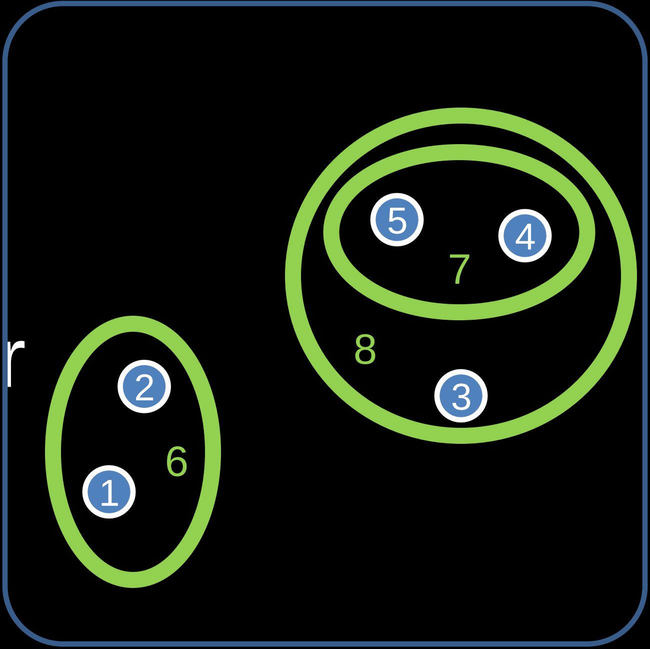
## The UPGMA process

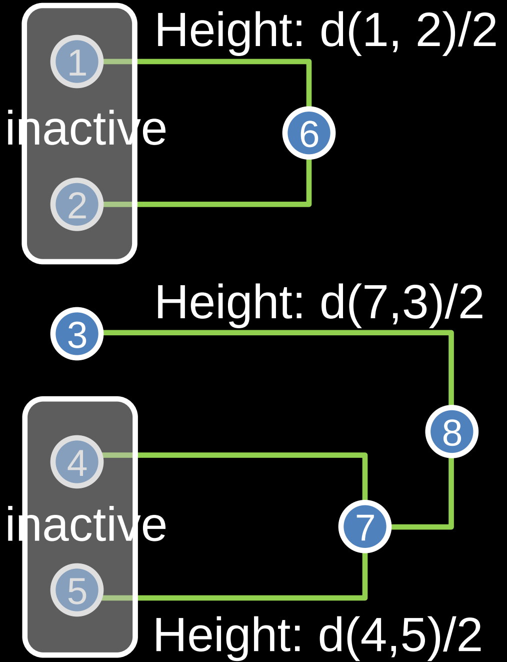
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (3,7)
- Step 3) Group the closest pair in a new cluster (8)
- Step 4) Set height of hypothetical shared ancestor at d(3,7)/2





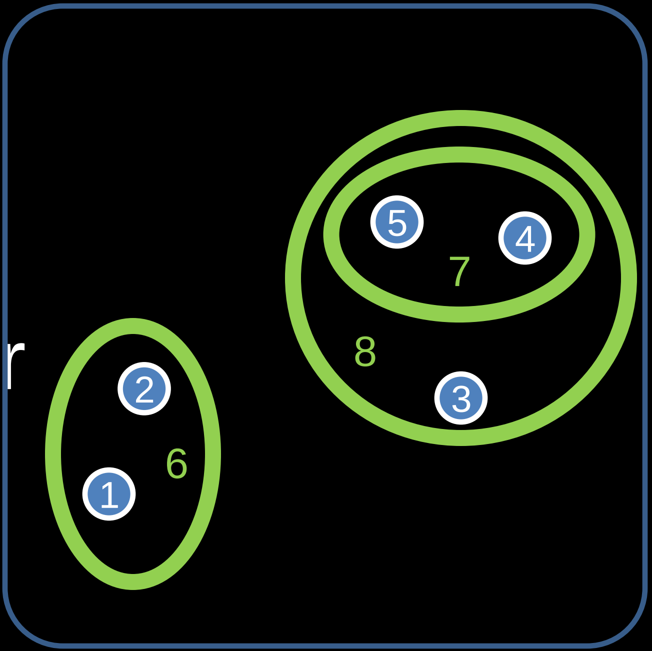
# The UPGMA process

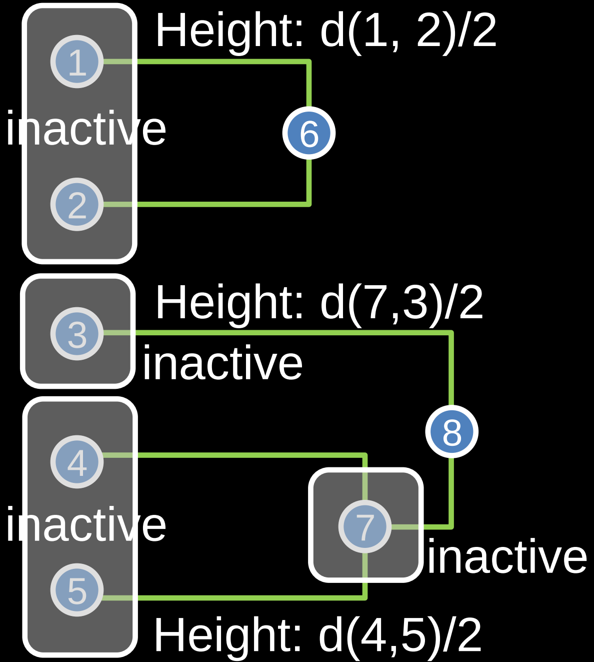
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (3,7)
- Step 3) Group the closest pair in a new cluster (8)
- Step 4) Set height of hypothetical shared ancestor at d(3,7)/2
- Step 5) Remove the old clusters 3,7, add the new one (8) to list of active clusters





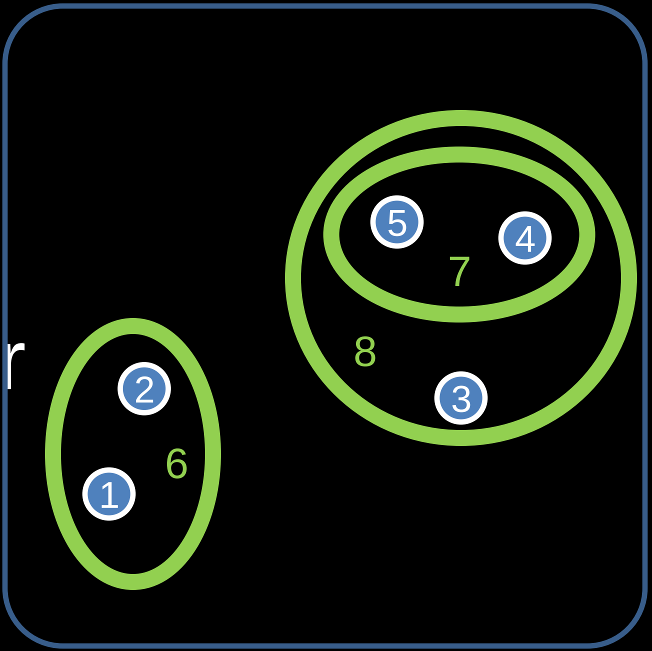
## The UPGMA process

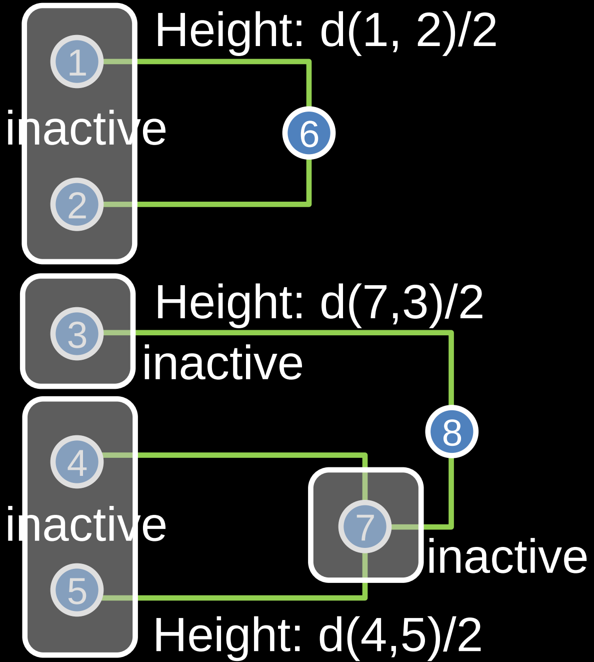
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

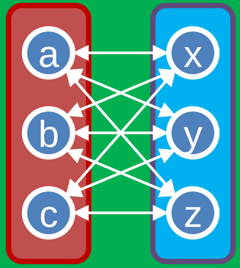
While (num active clusters > 1)

Step 2) Find the new closest pair among the active clusters (6,8)





The distance between two clusters is the average of the pairwise distances



Distance((abc), (xyz)) is the average of d(a,x) d(a,y), d(a,z), d(b,x), d(b,y), d(b,z), d(c,x), d(c,y), d(c,z)

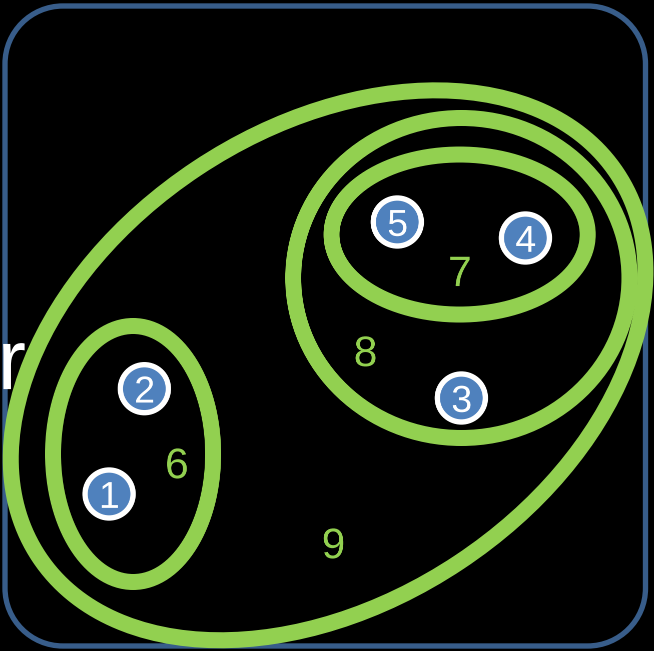
# The UPGMA process

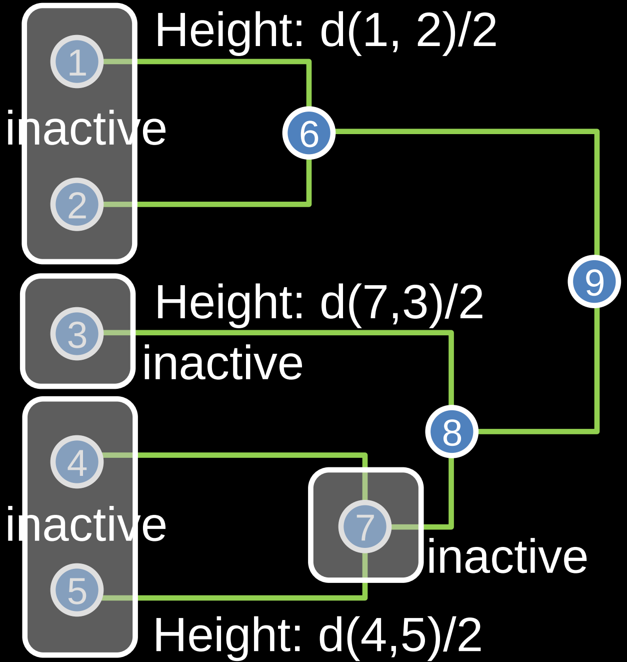
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (6,8)
- Step 3) Group the closest pair in a new cluster (9)





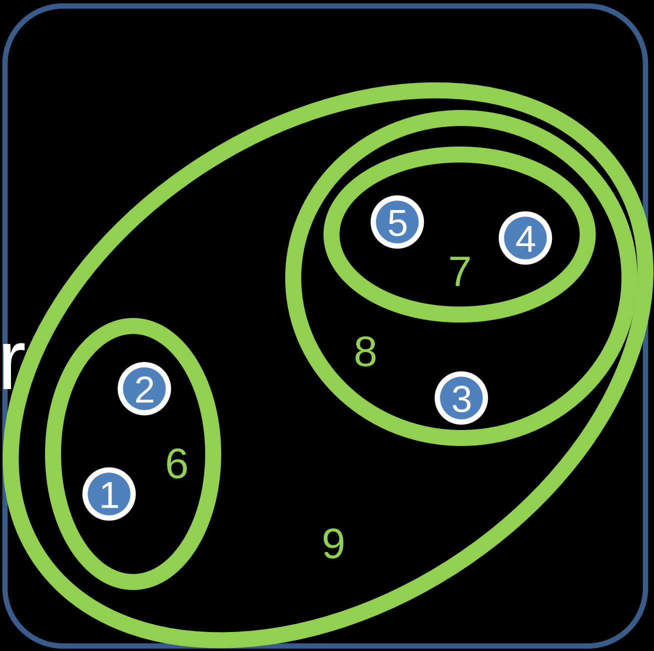
# The UPGMA process

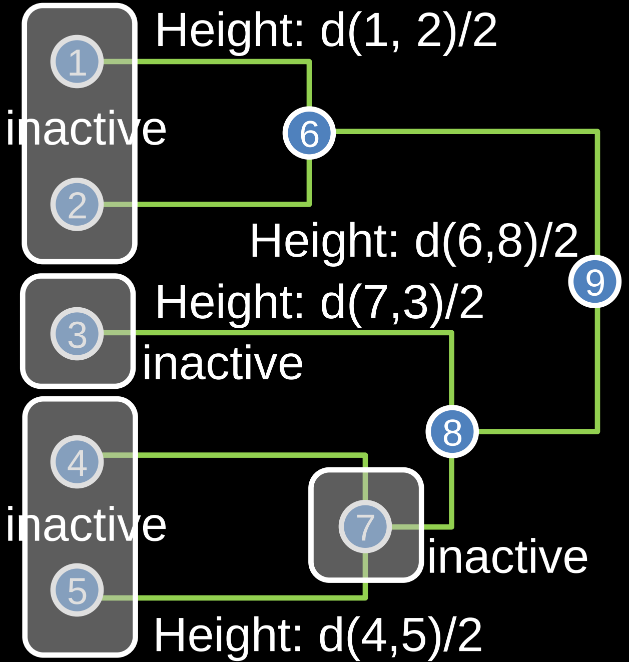
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (6,8)
- Step 3) Group the closest pair in a new cluster (9)
- Step 4) Set height of hypothetical shared ancestor at d(6,8)/2





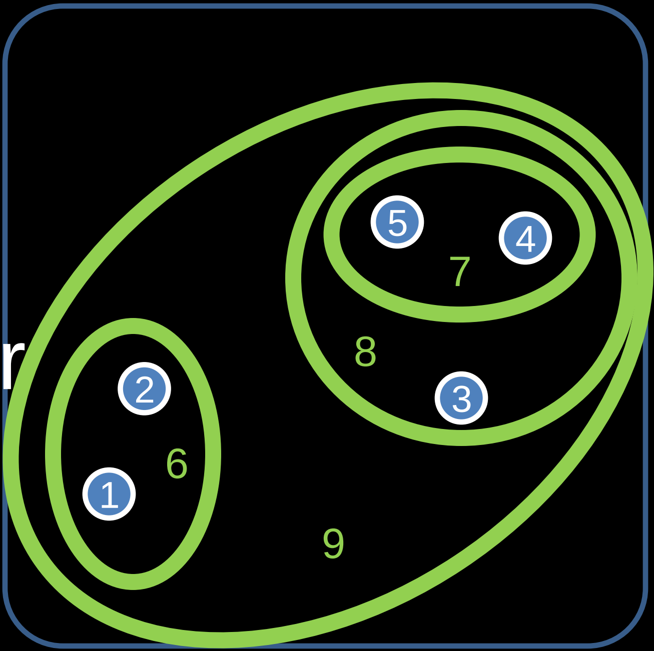
# The UPGMA process

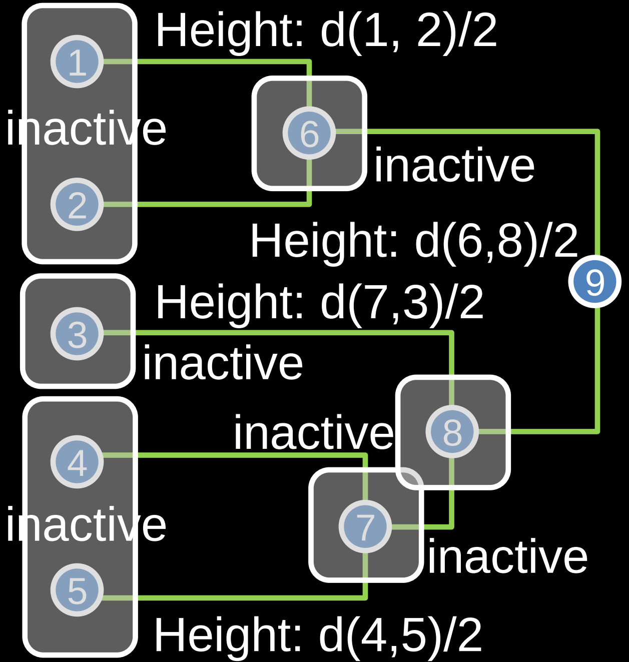
Input: A distance matrix

Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

- Step 2) Find the new closest pair among the active clusters (6,8)
- Step 3) Group the closest pair in a new cluster (9)
- Step 4) Set height of hypothetical shared ancestor at d(6,8)/2
- Step 5) Remove the old clusters 6,8, add the new one (9) to list of active clusters





# The UPGMA process

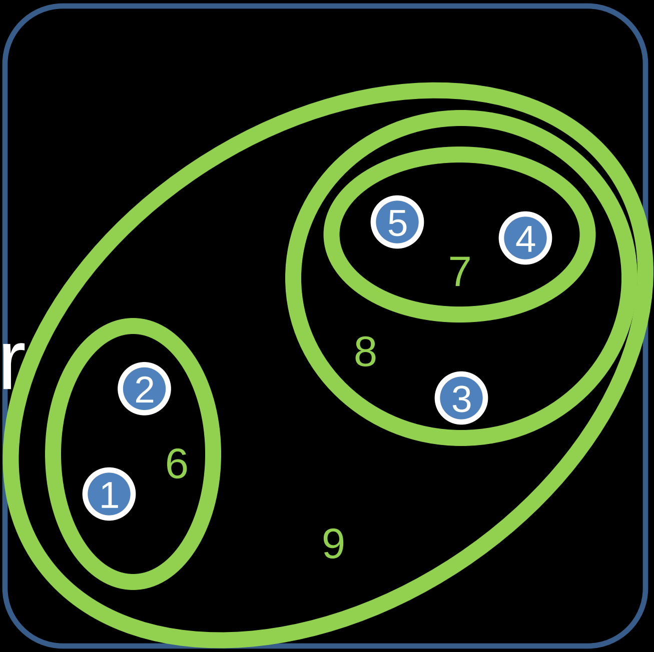
Input: A distance matrix

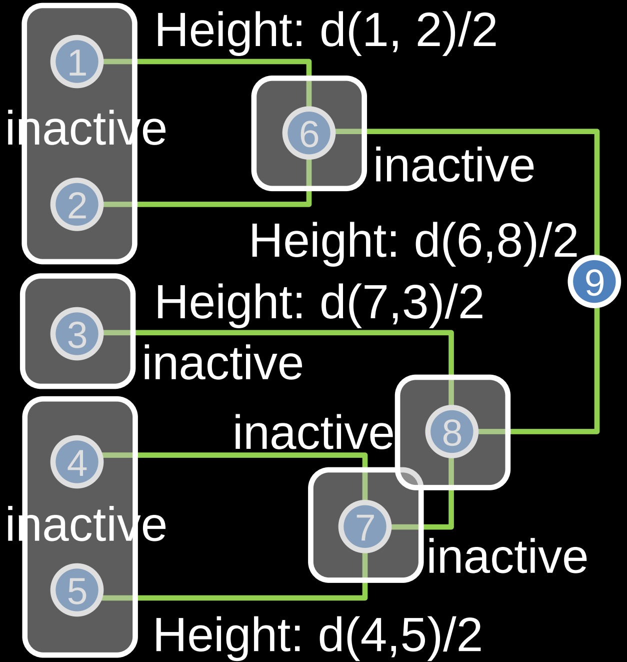
Step 1) make each sequence a member of its own active cluster

While (num active clusters > 1)

    Num active clusters = 1

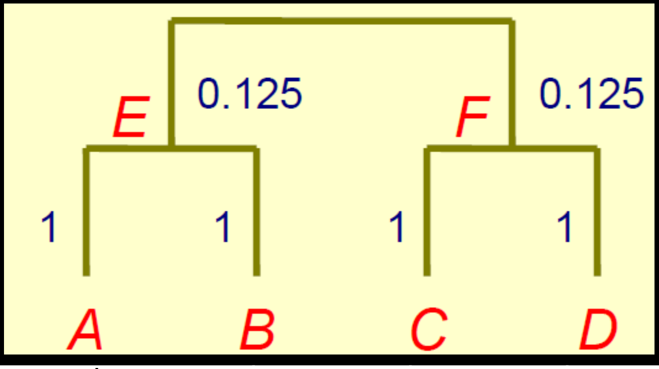
    Done.





# Suboptimality of UPGMA

- Recall that not every distance matrix is additive
  - Non-additive matrices cannot be converted into a tree such that the total edge weight between any two nodes is the same as distance in the matrix
- UPGMA attempts to remedy this by generating a similar tree
  - But it's not the same



UPGMA tree

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 2 | 2 | 2 |
| B | 2 | 0 | 3 | 2 |
| C | 2 | 3 | 0 | 2 |
| D | 2 | 2 | 2 | 0 |

Original Distance Matrix

Matrix computed with UPGMA tree

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 2 | 2.25 | 2.25 |
| B | 2 | 0 | 2.25 | 2.25 |
| C | 2.25 | 2.25 | 0 | 2 |
| D | 2.25 | 2.25 | 2 | 0 |

# Rules for a better tree

- A better tree should have lower total edge weight

  Why: Occam's Razor

  - Given two trees with different edgeweights, the lighter tree requires less random substitution to occur
  - It is more plausible that less random substitution occurs than more random substitution

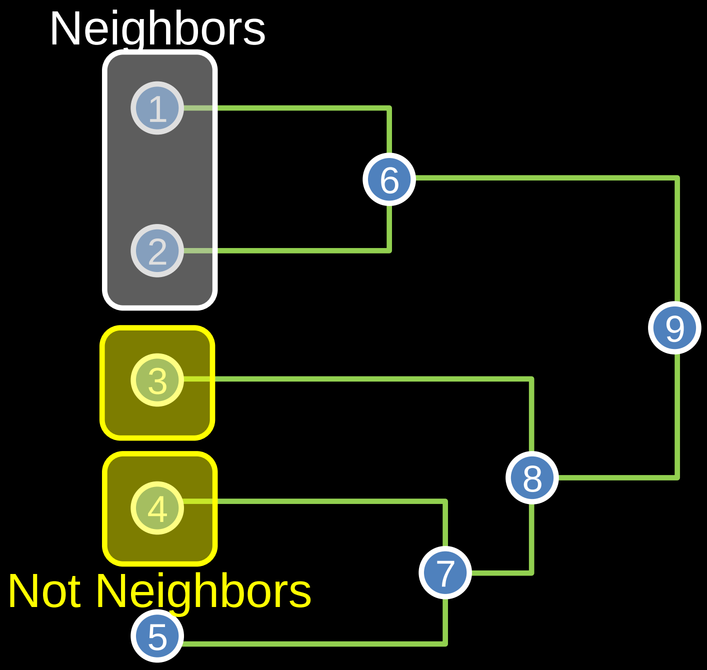
- Thus, all tree generation methods seek to reduce the total edge weight of a tree.

## Overview of Neighbor Joining

- Neighbor Joining tracks total tree weight in order to keep it minimal

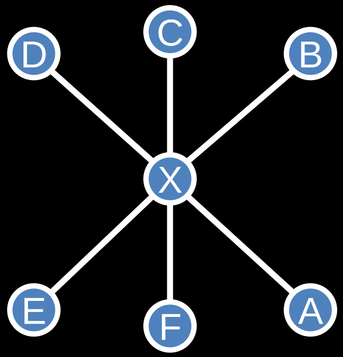
Definition: *Neighbor*

- Nodes x and y are said to be neighbors if they are separated in the tree by exactly one node



# Edgeweights in Neighbor Joining

Initial "Star" tree. Nodes correspond to sequences. N total nodes. X is extra.



Distance Matrix D, values called $D_{ij}$

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | | | | | |
| B | 5 | 0 | | | | |
| C | 4 | 7 | 0 | | | |
| D | 7 | 10 | 7 | 0 | | |
| E | 6 | 9 | 6 | 5 | 0 | |
| F | 8 | 11 | 8 | 9 | 8 | 0 |

- Neighbor joining regards the initial state of the tree as a "star"
- The sum $S_0$ of the edge weights is written:

$$S_O = \sum_{i=1}^{N} L_{iX} = \frac{1}{N-1} \sum_{i<j} D_{ij},$$

- Each weight $D_{ij}$ corresponds to two legs of the star, since the path from i to j passes through X
- Hence, the sum of all $D_{ij}$ adds each leg N-1 times.
- Use this equation to get the weight of an individual edge, $L_{ix}$

# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Distance Matrix N=6

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | | | | | |
| B | 5 | 0 | | | | |
| C | 4 | 7 | 0 | | | |
| D | 7 | 10 | 7 | 0 | | |
| E | 6 | 9 | 6 | 5 | 0 | |
| F | 8 | 11 | 8 | 9 | 8 | 0 |

$L_{AU} = (5+4+7+6+8)/4 = 7.5$

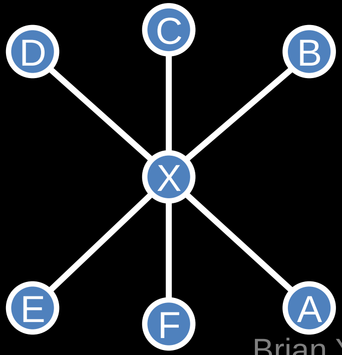
$L_{BU} = (5+7+10+9+11)/4 = 10.5$

$L_{CU} = (4+7+7+6+8)/4 = 8$

$L_{DU} = (7+10+7+5+9)/4 = 9.5$

$L_{EU} = (6+9+6+5+8)/4 = 8.5$

$L_{FU} = (8+11+8+9+8)/4 = 11$



# Neighbor Joining

Distance Matrix N=6

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | | | | | |
| B | 5 | 0 | | | | |
| C | 4 | 7 | 0 | | | |
| D | 7 | 10 | 7 | 0 | | |
| E | 6 | 9 | 6 | 5 | 0 | |
| F | 8 | 11 | 8 | 9 | 8 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

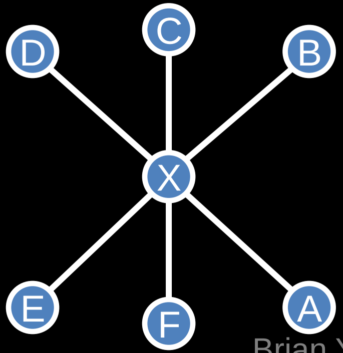
$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

$L_{AU} = (5+4+7+6+8)/4 = 7.5$
$L_{BU} = (5+7+10+9+11)/4 = 10.5$
$L_{CU} = (4+7+7+6+8)/4 = 8$
$L_{DU} = (7+10+7+5+9)/4 = 9.5$
$L_{EU} = (6+9+6+5+8)/4 = 8.5$
$L_{FU} = (8+11+8+9+8)/4 = 11$

This finds the closest pair of neighbors. Since any pair of nodes that touches the center node, U, can be neighbors, you must check all pairs of nodes adjacent to U.

Smallest $M_{ij}$: (We pick $M_{AB}$)
$M_{AB} = 5 - 7.5 - 10.5 = -13$
$M_{DE} = 5 - 9.5 - 8.5 = -13$

Resolve Ties arbitrarily. Here we pick $M_{AB}$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} – L_{iU} – L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U1 so that

$$L_{iU1} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

Note: weight (U,U1) does not need to be explicitly assigned

Distance Matrix N=6

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | | | | | |
| B | 5 | 0 | | | | |
| C | 4 | 7 | 0 | | | |
| D | 7 | 10 | 7 | 0 | | |
| E | 6 | 9 | 6 | 5 | 0 | |
| F | 8 | 11 | 8 | 9 | 8 | 0 |

$L_{AU} = (5+4+7+6+8)/4 = 7.5$

$L_{BU} = (5+7+10+9+11)/4 = 10.5$

$L_{CU} = (4+7+7+6+8)/4 = 8$

$L_{DU} = (7+10+7+5+9)/4 = 9.5$

$L_{EU} = (6+9+6+5+8)/4 = 8.5$

$L_{FU} = (8+11+8+9+8)/4 = 11$

Smallest $M_{ij}$: (We pick $M_{AB}$)

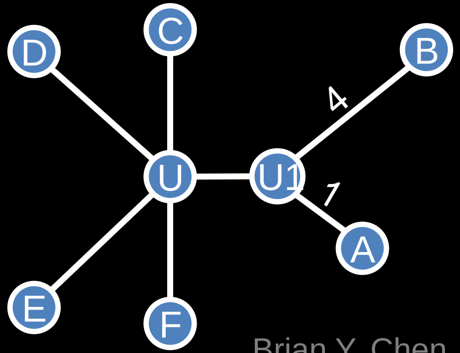
$M_{AB} = 5 – 7.5 – 10.5 = -13$

$M_{DE} = 5 – 9.5 – 8.5 = -13$

In this case:

$$L_{AU1} = D_{AB}/2 + (L_{AU}-L_{BU})/2 = 1$$

$$L_{BU1} = D_{BA}/2 + (L_{BU}-L_{AU})/2 = 4$$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U1 so that

$$L_{iU1} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

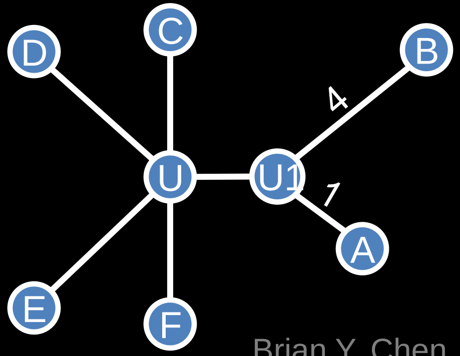
Step 4) Recalculate the distance matrix for all taxa x to U1, where

$$D_{xU1} = (D_{ix} + D_{jx} - D_{ij})/2$$

and i and j are the pair selected above

Distance Matrix N=5

| | U1 | C | D | E | F |
|---|---|---|---|---|---|
| U1 | 0 | | | | |
| C | 3 | 0 | | | |
| D | 6 | 7 | 0 | | |
| E | 5 | 6 | 5 | 0 | |
| F | 7 | 8 | 9 | 8 | 0 |



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Distance Matrix N=5

| | U1 | C | D | E | F |
|---|---|---|---|---|---|
| U1 | 0 | | | | |
| C | 3 | 0 | | | |
| D | 6 | 7 | 0 | | |
| E | 5 | 6 | 5 | 0 | |
| F | 7 | 8 | 9 | 8 | 0 |

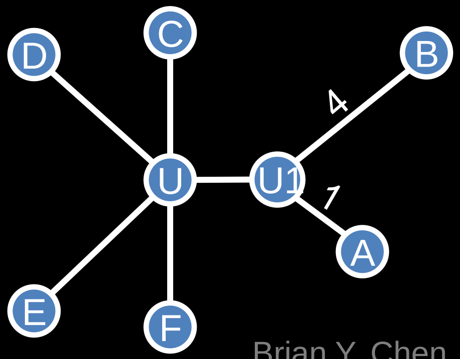
$L_{U1U} = (3+6+5+7)/3 = 7$

$L_{CU} = (3+7+6+8)/3 = 8$

$L_{DU} = (6+7+5+9)/3 = 9$

$L_{EU} = (5+6+5+8)/3 = 8$

$L_{FU} = (6+8+9+8)/3 = 10.6$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} – L_{iU} – L_{jU}$$

This finds the closest pair of neighbors. Since any pair of nodes that touches the center node, U, can be neighbors, you must check all pairs of nodes adjacent to U.

Resolve Ties arbitrarily. Here we pick $M_{DE}$

Distance Matrix N=5

| | U1 | C | D | E | F |
|---|---|---|---|---|---|
| U1 | 0 | | | | |
| C | 3 | 0 | | | |
| D | 6 | 7 | 0 | | |
| E | 5 | 6 | 5 | 0 | |
| F | 7 | 8 | 9 | 8 | 0 |

$L_{U1U} = (3+6+5+7)/3 = 7$

$L_{CU} = (3+7+6+8)/3 = 8$

$L_{DU} = (6+7+5+9)/3 = 9$

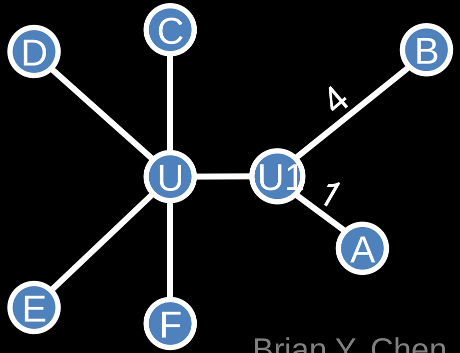
$L_{EU} = (5+6+5+8)/3 = 8$

$L_{FU} = (6+8+9+8)/3 = 10.6$

Smallest $M_{ij}$: (We pick $M_{DE}$)

$M_{CU1} = 3 – 7 – 8 = -12$

$M_{DE} = 5 – 9 – 8 = -12$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} – L_{iU} – L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U2 so that

$$L_{iU2} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

Note: weight (U,U2) does not need to be explicitly assigned

Distance Matrix N=5

| | U1 | C | D | E | F |
|---|---|---|---|---|---|
| U1 | 0 | | | | |
| C | 3 | 0 | | | |
| D | 6 | 7 | 0 | | |
| E | 5 | 6 | 5 | 0 | |
| F | 7 | 8 | 9 | 8 | 0 |

$L_{U1U} = (3+6+5+7)/3 = 7$

$L_{CU} = (3+7+6+8)/3 = 8$

$L_{DU} = (6+7+5+9)/3 = 9$

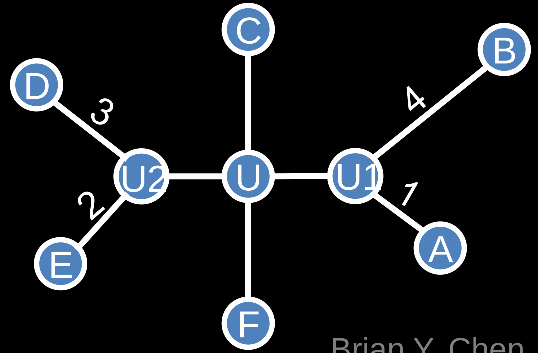
$L_{EU} = (5+6+5+8)/3 = 8$

$L_{FU} = (6+8+9+8)/3 = 10.6$

Smallest $M_{ij}$: (We pick $M_{DE}$)

$M_{CU1} = 3 – 7 – 8 = -12$

$M_{DE} = 5 – 9 – 8 = -12$



In this case:

$$L_{DU2} = D_{DE}/2 + (L_{DU}-L_{EU})/2 = 3$$

$$L_{EU2} = D_{DE}/2 + (L_{EU}-L_{DU})/2 = 2$$

# Neighbor Joining

Distance Matrix N=4

|  | U1 | C | U2 | F |
|---|---|---|---|---|
| U1 | 0 |  |  |  |
| C | 3 | 0 |  |  |
| U2 | 3 | 4 | 0 |  |
| F | 7 | 8 | 6 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

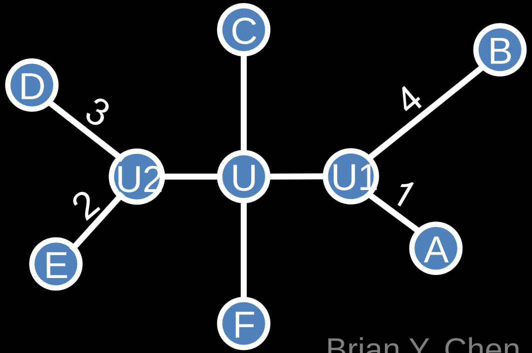
Step 3) Separate the pair (i,j) and create a new node U2 so that

$$L_{iU2} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

Step 4) Recalculate the distance matrix for all taxa x to U2, where

$$D_{xU1} = (D_{ix} + D_{jx} - D_{ij})/2$$

and i and j are the pair selected above



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Distance Matrix N=4

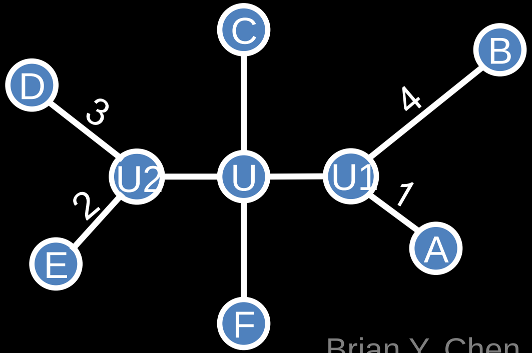
| | U1 | C | U2 | F |
|---|---|---|---|---|
| U1 | 0 | | | |
| C | 3 | 0 | | |
| U2 | 3 | 4 | 0 | |
| F | 7 | 8 | 6 | 0 |

$L_{U1U} = (3+3+7)/2 = 6.5$

$L_{CU} = (3+4+8)/2 = 7.5$

$L_{U2U} = (3+4+6)/2 = 6.5$

$L_{FU} = (7+8+6)/2 = 10.5$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

Distance Matrix N=4

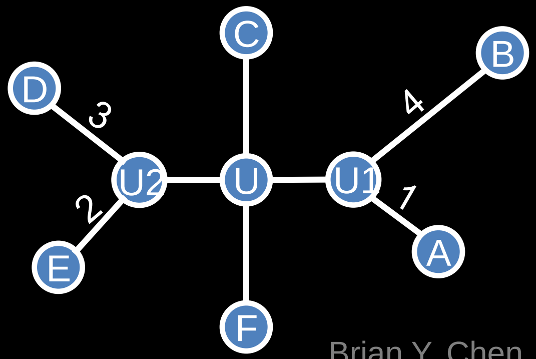
| | U1 | C | U2 | F |
|---|---|---|---|---|
| U1 | 0 | | | |
| C | 3 | 0 | | |
| U2 | 3 | 4 | 0 | |
| F | 7 | 8 | 6 | 0 |

$L_{U1U} = (3+3+7)/2 = 6.5$
$L_{CU} = (3+4+8)/2 = 7.5$
$L_{U2U} = (3+4+6)/2 = 6.5$
$L_{FU} = (7+8+6)/2 = 10.5$

This finds the closest pair of neighbors. Since any pair of nodes that touches the center node, U, can be neighbors, you must check all pairs of nodes adjacent to U.

Smallest $M_{ij}$: (No tie this time)
$M_{CU1} = 3 - 6.5 - 7.5 = -11$

Resolve Ties arbitrarily.
No ties this time



# Neighbor Joining

Distance Matrix N=4

| | U1 | C | U2 | F |
|---|---|---|---|---|
| U1 | 0 | | | |
| C | 3 | 0 | | |
| U2 | 3 | 4 | 0 | |
| F | 7 | 8 | 6 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U3 so that

$$L_{iU3} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

$L_{U1U} = (3+3+7)/2 = 6.5$

$L_{CU} = (3+4+8)/2 = 7.5$

$L_{U2U} = (3+4+6)/2 = 6.5$

$L_{FU} = (7+8+6)/2 = 10.5$

Smallest $M_{ij}$: (No tie this time)

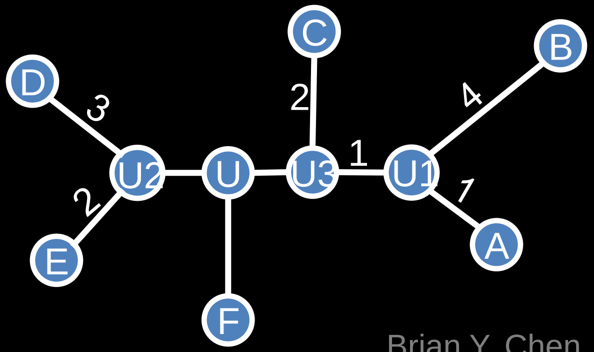
$M_{CU1} = 3 - 6.5 - 7.5 = -11$

Note: weight (U,U3) does not need to be explicitly assigned

In this case:

$$L_{CU3} = D_{CU1}/2 + (L_{CU}-L_{U1U})/2 = 2$$

$$L_{U1U3} = D_{CU1}/2 + (L_{U1U}-L_{CU})/2 = 1$$



# Neighbor Joining

Distance Matrix N=3

|  | U3 | U2 | F |
|---|---|---|---|
| U3 | 0 |  |  |
| U2 | 2 | 0 |  |
| F | 6 | 6 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

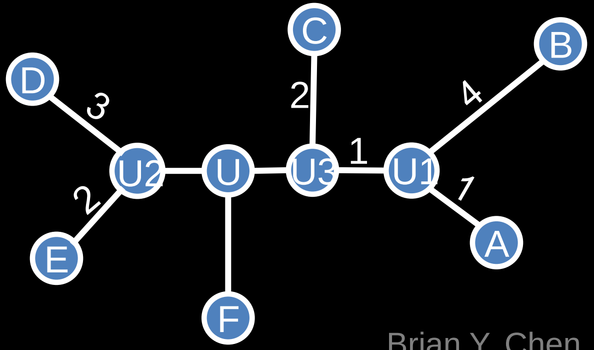
Step 3) Separate the pair (i,j) and create a new node U3 so that

$$L_{iU3} = D_{ij}/2 + (L_{iU} - L_{jU})/2$$

Step 4) Recalculate the distance matrix for all taxa x to U3, where

$$D_{xU1} = (D_{ix} + D_{jx} - D_{ij})/2$$

and i and j are the pair selected above



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

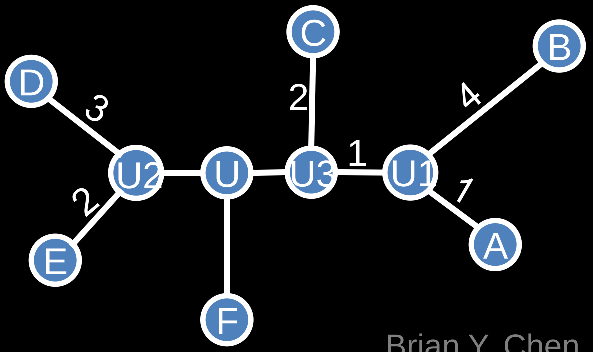
Distance Matrix N=3

| | U3 | U2 | F |
|---|---|---|---|
| U3 | 0 | | |
| U2 | 2 | 0 | |
| F | 6 | 6 | 0 |

$L_{U2U} = (2+6)/1 = 8$

$L_{U3U} = (2+6)/1 = 8$

$L_{FU} = (6+6)/1 = 12$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

This finds the closest pair of neighbors. Since any pair of nodes that touches the center node, U, can be neighbors, you must check all pairs of nodes adjacent to U.

Resolve Ties arbitrarily. We choose $M_{U2U3}$

Distance Matrix N=3

| | U3 | U2 | F |
|---|---|---|---|
| U3 | 0 | | |
| U2 | 2 | 0 | |
| F | 6 | 6 | 0 |

$L_{U2U} = (2+6)/1 = 8$

$L_{U3U} = (2+6)/1 = 8$

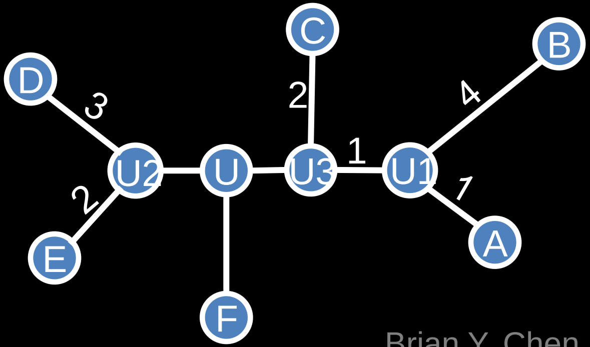
$L_{FU} = (6+6)/1 = 12$

Smallest $M_{ij}$: (We pick $M_{U2U3}$)

$M_{U2F} = 6 - 8 - 12 = -14$

$M_{U3F} = 6 - 7 - 12 = -14$

$M_{U2U3} = 2 - 8 - 8 = -14$



# Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} – L_{iU} – L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U4 so that

$$L_{iU4} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

Note: weight (U,U4) does not need to be explicitly assigned

Distance Matrix N=3

| | U3 | U2 | F |
|---|---|---|---|
| U3 | 0 | | |
| U2 | 2 | 0 | |
| F | 6 | 6 | 0 |

$L_{U2U} = (2+6)/1 = 8$

$L_{U3U} = (2+6)/1 = 8$

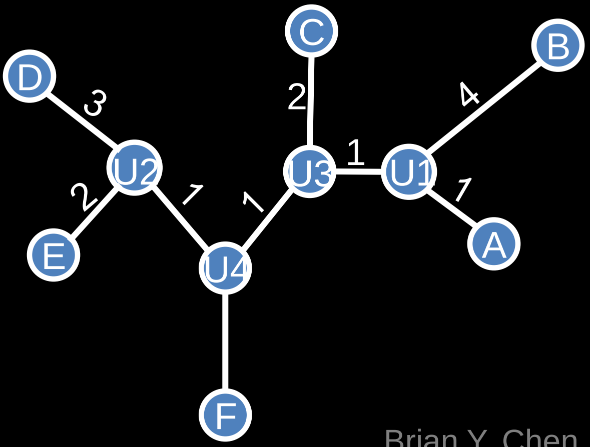
$L_{FU} = (6+6)/1 = 12$

Smallest $M_{ij}$: (We pick $M_{U2U3}$)

$M_{U2F} = 6 – 8 – 12 = -14$

$M_{U3F} = 6 – 7 – 12 = -14$

$M_{U2U3} = 2 – 8 – 8 = -14$



<u>In this case:</u>

$$L_{U2U4} = D_{U2U3}/2 + (L_{U2U}-L_{U3U})/2 = 1$$

$$L_{U3U4} = D_{U2U3}/2 + (L_{U3U}-L_{U2U})/2 = 1$$

# Neighbor Joining

Distance Matrix N=2

| | U4 | F |
|---|---|---|
| | | |
| | | |
| | | |
| U4 | 0 | |
| | | |
| F | 5 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} – L_{iU} – L_{jU}$$

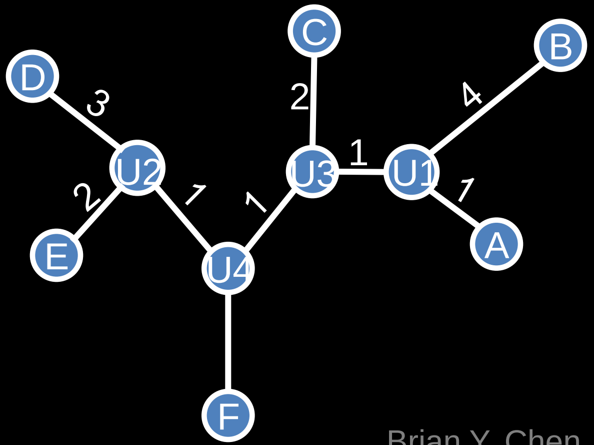
Step 3) Separate the pair (i,j) and create a new node U4 so that

$$L_{iU4} = D_{ij}/2 + (L_{iU}-L_{jU})/2$$

Step 4) Recalculate the distance matrix for all taxa x to U3, where

$$D_{xU1} = (D_{ix} + D_{jx} – D_{ij})/2$$

and i and j are the pair selected above



## Neighbor Joining

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

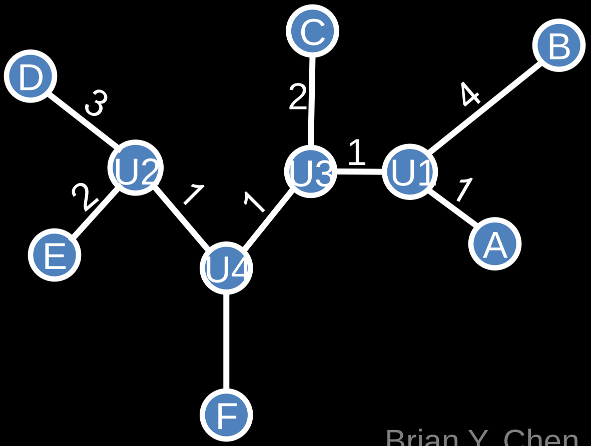
$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

This time computing this value is not possible, because N-2 = 0

Detecting this indicates the end of the algorithm

Distance Matrix N=2

| | U4 | F |
|---|---|---|
| U4 | 0 | |
| F | 5 | 0 |



# Neighbor Joining

Distance Matrix
N=2

| | U4 | F |
|---|---|---|
| | | |
| | | |
| U4 | 0 | |
| | | |
| F | 5 | 0 |

Input: distance matrix, star graph

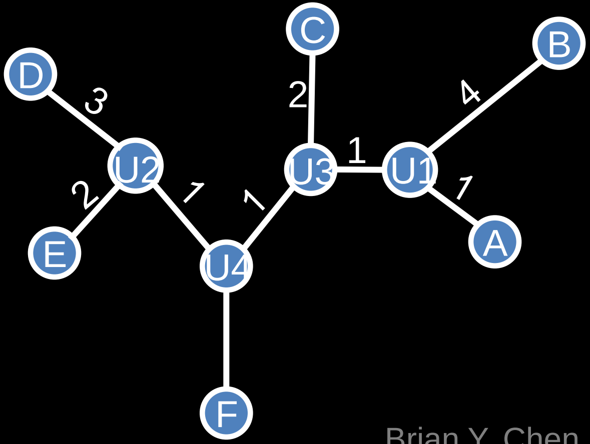
Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

Again, not possible. There is only one possibility for i or j.
Also, $L_{iU}$ is undefined for any i and j



# Neighbor Joining

Distance Matrix N=2

| | U4 | F |
|---|---|---|
| U4 | 0 | |
| F | 5 | 0 |

Input: distance matrix, star graph

Step 1) estimate Star leg weights $L_{iU}$

$$L_{iU} = (\text{Sum of all } D_{ij})/(N-2)$$

Step 2) Find pair (i, j) with smallest $M_{ij}$,

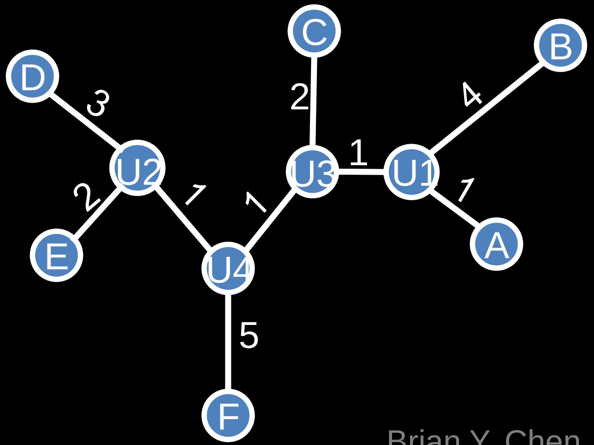
$$M_{ij} = D_{ij} - L_{iU} - L_{jU}$$

Step 3) Separate the pair (i,j) and create a new node U4 so that

$$\cancel{L_{iU4} = D_{ij}/2 + (L_{iU}-L_{jU})/2}$$

$$L_{iU4} = D_{FU4}$$

There is no pair (i,j) to consider, so we do not need to worry about how the tree will be split: so we just use the Distance Matrix, $D_{FU4}$



# Neighbor Joining

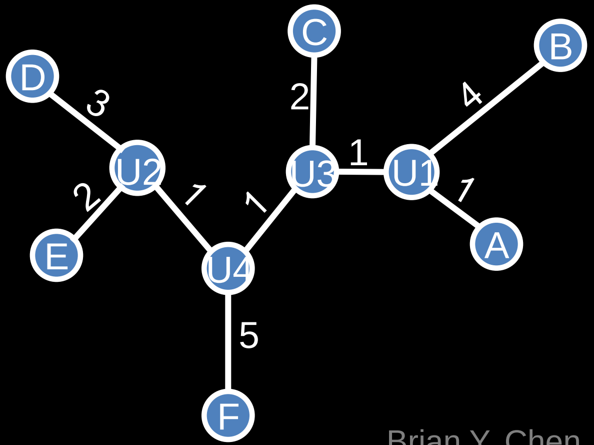
Input: distance matrix, star graph

Output:

Tree topology and Edge Weights

Distance Matrix N=2

| | U4 | F |
|---|---|---|
| U4 | 0 | |
| F | 5 | 0 |



Questions